Monitoring an anonymity network: toward the deanonymization of hidden services

Marco Simioni[a,b], Pavel Gladyshev[a], Babak Habibnia[a], Paulo Roberto Nunes de Souza[c]

[a] *Digital Forensics Investigation Research Laboratory, University College Dublin*
[b] *IBM Research, IBM Ireland*
[c] *Department of Computing, Universidade Federal do Espírito Santo*

Abstract

Anonymity networks are an example of Privacy Enhancing Technology (PET) whose historical goal is to avoid censorship, preserve users privacy, and promote freedom of speech. Such networks, however, also provide a “safe haven” for criminal activity: previous research observed a dominance of commerce platforms delivered as hidden services within The Onion Router (Tor) network, undoubtedly the most popular anonymization technology at the time of writing, largely around narcotics and illegal financial services.

Extensive research has been conducted on locating hidden services on the Tor network, but a general method that is able, given a service delivered via anonymity network, to effectively produce a list of candidate nodes responsible for delivering the service still remains an open research problem. In this paper we describe the infrastructure we have designed and implemented for monitoring the Invisible Internet Project (I2P) network, which is a smaller scale anonymity network compared to Tor but already proven to be used for illicit activities, and how its output can be used to enable such general method.

Keywords— anonymity networks, i2p, tor, deanonymization

1. Introduction

The main goal of Privacy Enhancing Technologies[1] such as anonymity systems is preserving user privacy, promoting freedom of speech, and facilitating the free flow of information. However, the level of illegal and criminal activities within these networks gives a cause for concern[2]. When such crimes are committed, the goal of an investigator is to determine who the criminal is, and network connection information is paramount in order to identify what occurred. The relationship between the “IP address”, the “port number”, and the “time stamp”, in presence of a crime, is the most common clue used to locate an offender’s workstation or server, and can help in revealing the location of the offender[3].

However, thanks to the adoption of anonymity networks for delivering their services, not only the confidentiality of the offender’s communication is protected by multiple layers of encryption, but network information such as the ultimate recipient of a communication is obfuscated by a special routing mechanism. Locating a target such as a web site that makes use of an anonymity network is then a non-trivial task and solely analyzing the traffic data (i.e. the content of IP packets) and metadata (e.g. sender address and receiving address of IP packets) is not sufficient, due to the above mentioned factors.

Consider a website delivered via anonymity network, and an observer who needs to identify which IP address this service is delivered from. Our approach consist of measuring the availability of all the nodes connected to an anonymity network over time as a sequence of 0s and 1s along with their IP address, while at the same time measuring the availability of a targeted web service as a sequence of 0s and 1s. There is reasonable confidence that that the node delivering the web service is among the set of nodes whose availability sequence resembles the availability sequence of the target.

The present work is organised as follows. Background material is presented in Section 2. Section 3 introduces the Invisible Internet Project (I2P) anonymity network, and its related concepts. In Section 4 the infrastructure we have designed for monitoring I2P is described. In Section 5 and 6 we discuss the experiments we have conducted and their outcome.

2. Background

2.1. Anonymity

While a variety of definitions of the term *anonymity* —derived from the Greek word ἀνωνυμία, anonymia, meaning ”without a name” or ”namelessness” —have been suggested, this work will use the definition suggested by [5]: ”Anonymity is the state of being not identifiable within a set of subjects, the anonymity set”. Other simplified models of anonymity can also be used when describing systems providing anonymity, as proposed by [6]. For instance, consider three users Alice, Bob, and Charlie, who are communicating through the usage of a system—such as an anonymity network—and an attacker Oscar, who wants to identify who is communicating with whom. Such system is said to provide anonymity if Oscar after attacking the system learns no new information to confirm or deny who Alice is communicating with.

2.2. Deanonymization

For the purposes of this research, we define the deanonymization of a service—e.g. a website—delivered via anonymity network as the ability to discover its IP address and port number at a specific point in time. Within the context of anonymity networks such information is paramount to the investigator, enabling the further application of different known techniques on the reduced set, and eventually police work to be conducted on the target in order to locate it.

This is a non-trivial task, as such networks provide not only confidentiality to the transmitted messages but also anonymity to the nodes that participate in a communication. It is therefore acceptable to also define deanonymization as the ability to reduce the broad anonymity set to a smaller set of potential candidate nodes.

2.3. Hamming Distance

Throughout this work, the Hamming Distance will be utilized for comparing the availability over time between two nodes or between a node and a service of an anonymity network. The availability of a node—or a service—can be expressed as a bit array, where each bit represents a time instant, and 0 indicates the the node—or the service—is not available or no information about it is known at the specific instant, while 1 indicates the the node—or the service—is available. The Hamming Distance (HD) on two generic bit arrays $u[k]$ and $v[k]$ of equal length can be defined as the proportion of disagreeing components:

$$HD = \frac{c_{01} + c_{10}}{K}$$

with c_{ij} being the number of occurrences where $u[k] = i$ and $v[k] = j$ for $k < K$, and K is the length of the bit array, usually equal to 1440 when comparing daily observations having a sampling interval equal to 1 minute.

3. The Invisible Internet Project (I2P)

Tor[4] is undoubtedly the most popular and well understood anonymization technology, and recent events suggest that extensive work has been done by law enforcement agencies to successfully hinder criminal organizations who rely on the Tor network[12]. However, a number of alternative networks to Tor exist[13]. Similarly, The Invisible Internet Project (I2P)[7] is an anonymity network technology that overlays normal packet-switched computer networks, such as the Internet. I2P evolved in parallel with the Tor project and avails of features that make it more resilient to certain deanonymization attacks. Unlike Tor, which provides static bi-directional anonymous communication channels between two applications, I2P provides a generic packet-switched network layer like the Internet Protocol. The communicating endpoints in I2P networks are identified by their cryptographic identities instead of IP addresses, and several transport and application layer protocols are implemented on top of I2P. Applications that communicate via I2P are doing so via I2P tunnels.

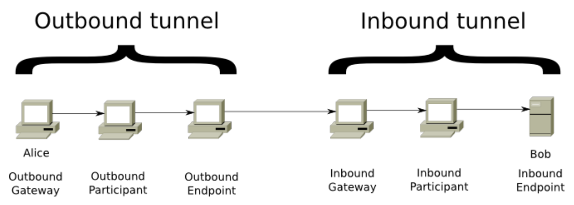

Figure 1: I2P tunnels, source https://geti2p.net/en/docs/how/tech-intro

The key concepts of I2P tunnels are illustrated in Figure 1. Suppose that the web browser of Alice wants to anonymously fetch a web page from the web server owned by Bob. To achieve that, the web server must first create an inbound tunnel, which consists of an inbound gateway followed by a chain of I2P nodes randomly chosen from the I2P network. I2P nodes are functionally similar to Tor relay nodes, and their network contacts such as IP address and port number are distributed via data structures named *RouterInfo*. At the end of the inbound tunnel there usually is a node, also called the inbound endpoint, that is delivering a service such as as website. Unlike Tor communication channels, the tunnel expires after a period of time, and a new tunnel needs to be created to replace it

To enable other applications to reach the service, its “contact information” needs to be published. This contact information is called *LeaseSet* and it documents the tunnel identifier together with its expiration time and other related information. LeaseSets are published in the distributed peer-to-peer database called netDB. Due to the distributed nature of netDB, disrupting the operation of a particular server on I2P is non trivial, because there are no centralized catalogs that could be shutdown or sanitized upon request.

Like Bob’s server, Alice’s computer also needs to establish an I2P tunnel, the outbound tunnel. It begins at the Alice’s computer (the outbound endpoint), traverses a number of I2P nodes and terminates with the outbound gateway. In order to send I2P packets to Bob’s web service, Alice’s computer needs to find the LeaseSet of the web service in netDB and use the information contained in it for specifying the destination for the packet. The packet is then transmitted through the outbound tunnel to the outbound gateway. The outbound gateway forwards it to the inbound gateway of the corresponding inbound I2P tunnel that delivers it to the destination. To achieve anonymity I2P uses multi-layered encryption of data between the endpoint and the gateway, with a technique called *onion routing*[8], which ensures that every node in the tunnel only knows the identity of the node immediately preceding it or immediately following it. To prevent dishonest gateways from sniffing user data, an additional layer of end-to-end encryption is used between Alice’s web browser and Bob’s web server.

To mitigate the vulnerability to traffic analysis, a technique named *garlic routing* is also adopted: multiple packets from the endpoint to different destination are pooled and transmitted together from the endpoint to the gateway, with the latter sending them one by one to their respective destination. The usage of *garlic routing*, combined with the limited lifetime of tunnels and the unidirectional nature of the tunnels - HTTP request and HTTP response must arrive via different tunnels - are supposed to make deanonymization of I2P by traffic analysis harder than in the case of Tor network.

Despite these design anonymity enhancements, we believe that I2P still provides a lot of useful information for investigations and intelligence as discussed in the next sections.

4. Monitoring infrastructure

Achieving our goal of measuring the availability of all the nodes connected to an anonymity network and collecting their IP addresses is a non-trivial task. Due to the decentralized nature of anonymity networks, which makes them resilient to DoS attacks among other privacy-preserving benefits, there is no centralised database where the list of nodes with their respective IP addresses can be retrieved, and our goal can only be achieved with a dedicated monitoring infrastructure. In designing our monitoring infrastructure, we seek to answer the following question: *how can we acquire information such as IP address and port number of all the nodes connected to an anonymity network at any given point in time, without altering the standard behaviour of the anonymity network software?*

As previously described, in order to access these networks, a user is normally required to install and run specific software and it is not uncommon for the development teams of such networks to embed some form of metrics collection within the software itself, or alternatively to provide additional tools that volunteers have the choice to execute, in order to monitor the behaviour of the network as a whole and to react to potential threats.

For instance the Tor Project organization, primarily responsible for maintaining the software that enables the Tor anonymity network, launched the Tor Metrics project[9] for "monitoring, understanding, and improving the network", while clearly stating how "any metrics collected must not undermine the anonymity or security properties of the Tor network.". The Tor Project also provides[10] a detailed description of their architecture and the aggregated data they collect.

Conversely, one of the fundamental characteristics of I2P is that it relies on a distributed database, the *netDB*, for storing information regarding the infrastructure of the network itself such as IP addresses and port numbers of known I2P nodes, as described in section 3. The I2P network developers operate a website[11] where statistics from the *netDB* database are collected by a subset of nodes and aggregated and made available to the public. Such metrics are being shared by volunteers who, in order to enable such collection, choose to contribute to it by running along with their node a specific software[14]. By doing so, anonymous statistics are being collected by these nodes and then aggregated by a central server, and the stats.i2p website acts as the main dashboard where several aggregations are presented.

In order to implement our measurement system, we have decided to follow an approach similar to the one implemented by the I2P network developers. The ease of access to fragments of the netDB database make I2P appealing for such experiments, along with the contained size of the network, when compared to the size of the Tor network in terms of number of nodes and users. This section describes the architecture of our monitoring infrastructure.

4.1. Monitoring Infrastructure

At the time of writing, the stats.i2p [11] website reports the estimated average number of concurrent nodes connected to the I2P network of 62.657 within the last 3 months of observation as shown in Figure 2.

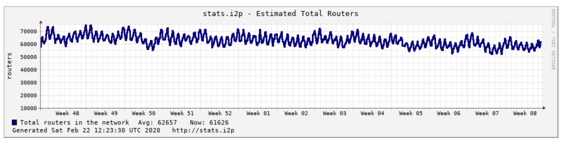

Figure 2: stats.i2p - 20200222 - Estimated Total Nodes

As described in section 3, in I2P the information we seek to monitor can be found in the *RouterInfo* data structure, which represents the contact information of a given node at a given point in time. These data structures are stored within the distributed database netDB, and such database is maintained by a subset of nodes connected to the network, the *floodfill* nodes: each of these *floodfill* nodes is responsible for maintaining a portion of the netDB. One possible way to achieve our goal would be to have an exhaustive snapshot of the netDB distributed database at any given point in time.

We have then designed a monitoring system as per Figure 3.

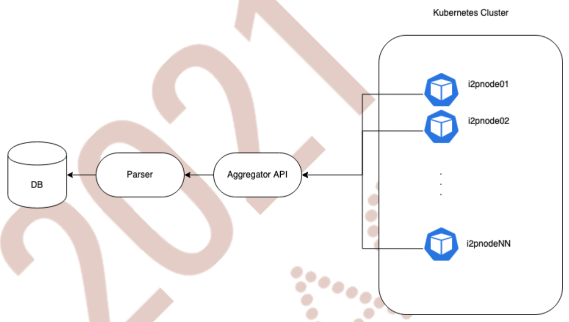

Figure 3: i2p monitoring architecture

The monitoring nodes (i2pnode01, i2pnode02...i2pnodeNN) are I2P *floodfill* nodes under our control that leverage an i2pd[15] daemon, formerly known as The PurpleI2P Project, a "full-featured C++ implementation of I2P client". Compared to the more popular and feature-rich Java implementation counterpart[16], i2pd is "less resource intensive, consuming less memory and CPU", and at the same time "has some major optimizations for faster cryptography which leads to less consumption of processor time and energy". These characteristics allowed us to deploy I2P *floodfill* nodes on Docker[17] containers built with an image having a small footprint of only 8MB, 3MB more than the 5MB base image Alpine Linux[18]. Only minor modifications have been implemented into the standard i2pd software in order to dump all the netDB information in scope of our experiment at regular intervals of 1 minute, and send a compressed version of such information to an Aggregator API endpoint. These containers have been deployed on Edge-net[20][21], a *"modern distributed edge cloud, incorporating advances in Cloud technologies over the past few years"*, which is built upon the Kubernetes[19] containers orchestration system. Edge-net allowed us to conduct measurements in a distributed manner, being the network constituted of nodes from a number of Universities: at the time of writing, the network is comprised of 38 geographically distributed nodes, primarily located in the US. The time syncronization of the nodes is left to the Operating System and maintained via NTP protocol[23]. For more details on the deployment infrastructure, the reader should be referred to the Edge-net and the Kubernetes documentations [19][21].

Each *floodfill* node provides a compressed binary dump containing information extracted from the portion of distributed database netDB that the node is responsible of maintaining. The compressed dump contains a timestamp representing the point in time when the information from the netDB has been extracted, in addition to the following attributes of for every node that the *floodfill* node have knowledge of:

- The IP address of the known node
- The Port number of the known node

5. Data collection analysis

Leveraging the monitoring infrastructure described in the previous section, two different experiments have been conducted:

- A first experiment, spanning multiple days, aimed at measuring how much visibility the infrastructure could obtain;
- A second experiment, made of multiple 1 or 2 days long experiments, with the goal of looking into the feasibility of deanonymization of a hidden node with only the collected data.

A detailed description of these experiments will be provided in this section.

5.1. Network visibility

As previously mentioned, the first experiment has been conducted in order to measure the visibility that the implemented infrastructure is able to obtain. By visibility, we define the number of nodes that our infrastructure have knowledge of at any given point in time, with respect of the estimates provided by the developers owned i2pstats dashboard.

From 2019-12-11 to 2019-12-17 inclusive, a number of monitoring *floodfill* nodes varying between 28 and 33 participated to the experiments, depending on cluster availability. According to stats.i2p, the estimated number of nodes connected to the network during this time averaged between 55k and 75k (see Figure 2, Weeks 50 and 51).

On average, a monitoring node deployed on our infrastructure had knowledge of 4752 nodes connected to the I2P network at any given minute. The total number of concurrent unique node identities observed varied from 8022 to 9971 with an average of 8890, as shown in Figure 4, or approximately 13%-15% of the network size estimated by stats.i2p. The second experiment will confirm that such visibility is sufficient to deanonymize a service delivered via anonymity network.

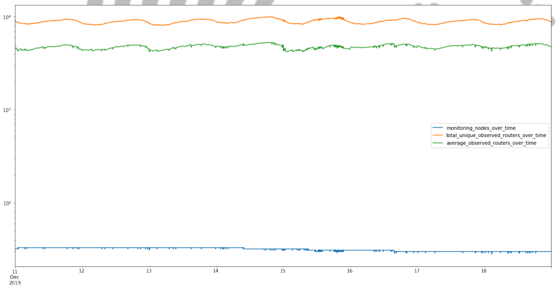

Figure 4: From 2019-12-11 to 2019-12-17 - Number of monitoring nodes vs. number of observed nodes

The total number of daily unique node identities observed by our infrastructure averaged 31.5k. With previous empirical studies[24], researchers were able to observe an average total number of 32k peers by controlling 20 nodes. Despite differences in the monitoring setup—notably, the ratio between floodfill vs non-floodfill monitoring nodes, and the resources allocated to each monitoring node—the results are aligned. More investigation is needed in order to understand what configuration could provide better visibility of the totality of the network.

5.2. Second experiment - deanonymization of a node

One of the primary goals of anonymity networks is to provide responder anonymity: a node should be able to deliver a service without having to disclose its IP address to the users of the service. Or alternatively, a user accessing a service delivered via anonymity network should not be able to obtain knowledge of the IP address of the node that delivers the service.

The second set of experiments aims at answering the following question: *without altering the standard behaviour of the I2P anonymity network software, and by analysing the nodes information observed in the netDB, is it possible to infer IP address and port number of a target service of interest delivered via anonymity network?*

Similarly to the experiment described in Section 5.1 a number of monitoring *floodfill* nodes have been deployed to the Edge-net Kubernetes cluster. An additional *non-floodfill* node t, or *the target node*, has been deployed on a dedicated server hosted under our control in a different location at the Universidade Federal do Espírito Santo, and a simple hello-world web page is served from this node as a website delivered via I2P. This section will describe the experiments that have been conducted and the results of the analysis of the netDB data.

5.2.1. Description

As previously described, by connecting to the I2P network every monitoring node m stores a portion of the netDB distributed database. With $R(t)$ being a set of unknown cardinality representing all the nodes connected to the I2P network at time t, at any given time t the portion of netDB stored by each monitoring node m contains information for a number of nodes, represented as $R^m(t) \subset R(t)$. Let the Availability of an I2P node defined as

$$A_r^m(t) = \begin{cases} 1 & \text{if RouterInfo is present} \\ 0 & \text{otherwise} \end{cases}$$

represents the availability at time t of information regarding the node r in the netDB portion maintained by the monitoring node m. Snapshots of these portions of netDB are taken by each monitoring node at regular times, such that

$$A_r^m[k] = A_r^m(kT)$$

represents the time sequence of samples at intervals T=1 minute, with $0 \leq k \leq 1439$ for each day of observation.

An average of 18 monitoring nodes were deployed on the Edge-net cluster, varying according to the nodes availability. The availability samples of a node r observed by each monitoring node m, $A_r^m[k]$, can be aggregated as:

$$A_r[k] = \bigvee_{m \in M} A_r^m[k]$$

which describes the availability of a node r being observed as online by at least one monitoring node of our monitoring infrastructure at the interval k.

For a given day, this availability can then be represented as a bit array with a length of 1440, such that:

$$A_r = [A_r[0], A_r[1] ... A_r[1438], A_r[1439]] = [0/1, 0/1 ... 0/1, 0/1]$$

The reader should note that the observed availability of a node may not exactly reflect its effective availability on the network: it is reasonable to assume that the more monitoring nodes an observer is able to deploy, the more accurate the observed availability will reflect the effective availability of the node. Furthermore, network congestion and netDB expiration timeouts, among other factors, may introduce noise in the observed availability.

During these experiments, specific patterns of online availability have been enforced by regularly enabling and disabling the i2pd software on the dedicated server that is delivering the *targeted* website via I2P anonymity network. The availability of the *target* node can be described as:

$$A_t[k] = \begin{cases} 1 & \text{if the I2P software on the target node is enabled} \\ 0 & \text{otherwise} \end{cases}$$

Represented in Figure 5, the patterns of online availability that have been enforced to the i2pd server hosting the website are defined as follows:

- During day 1 and 2, the i2pd daemon has been intermittently started and stopped every 4 hours, in order to generate cycles of 4 hours of continued online availability followed by 4 hours of offline absence.
- During day 3 and 4, the i2pd daemon has been intermittently started and stopped in order to generate cycles of 7 hours of continued online availability followed by 1 hour of offline absence.
- During day 5 and 6, the i2pd daemon has been intermittently started and stopped in order to generate cycles of 11.5 hours of continued online availability followed by 0.5 hour of offline absence.

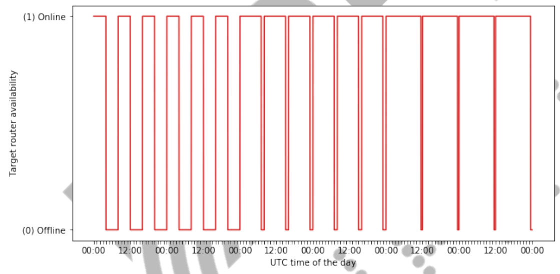

Figure 5: Online availability pattern injected over 6 days

In order to understand if the pattern injected into the availability of the target node can be observed by the monitoring infrastructure, on a given day the Hamming Distance can be calculated between each observed node availability $A_r[k] \forall r \in R$ and the known target availability $A_t[k]$:

$$HD_r = \frac{c_{01}^r + c_{10}^r}{n}$$

where c_{01}^r represents the number of occurrences where $A_r[k]$=0 and $A_t[k]$=1, while c_{10}^r represents the number of occurrences where $A_r[k]$=1 and $A_t[k]$=0 within the period of observation. In other words, the HD distance represents the proportion of samples where the observed availability r differs from the known effective availability of the target node t. Distances can

address	HD	HD_rank	cumulative
T.T.T.T	**0.075694**	**1**	**1**
a.a.a.a	0.188194	2	2
b.b.b.b	0.195139	3	4
c.c.c.c	0.195139	3	4
d.d.d.d	0.196528	4	5
...	...	...	...

Table 1: day 1 target node observation ranking (top 5 entries), 692 unique HD values and 41.311 observations

address	HD	HD_rank	cumulative
T.T.T.T	**0.095139**	**1**	**1**
e.e.e.e	0.166667	2	2
f.f.f.f	0.186111	3	3
g.g.g.g	0.186806	4	5
h.h.h.h	0.186806	4	5
...	...	...	...

Table 2: day 2 target node observation ranking (top 5 entries), 692 unique HD values and 42.571 observations

assume values in the range 0 to 1, where 0 represent nodes whose the observed availability resemble the availability of the target node under our control.

HD unique values calculated in a given day are sorted and a *HD rank* is assigned to each observed HD distance value, so that nodes with equal HD value have also same rank, and an HD distance ranks 1st if it is the miminum HD distance calculated for the given day. For a node r with a given distance HD_r, a cumulative number of nodes is also defined, as the total number of nodes that have a distance below or equal HD_r.

Results for day 1 and day 2 are shown[1] in Table 1 and 2.

With a value of 0.075694 and 0.095139 respectively in day 1 and day 2, the Hamming Distance calculated on the observation of the target node IP address[2] ranks first out of 692 unique values of Hamming Distance both in day 1 and day 2. Results suggest that the synthetic pattern injected on the availability of the target node during day 1 and day 2 can easily be detected by the observation infrastructure deployed, consisting of 18 nodes on average, each sampling netDB database at intervals of 1 minute each.

Figure 6 shows how the observed availability compares to the effective availability for day 1 and day 2. The reader should observe there are discrepancies between the observed availability and the effective availability. The delay between the offline-to-online transition at the source and the offline-to-online transition detected by the monitoring infrastructure is negligible. However, a delay of about 60 minutes appears evident between the online-to-offline transitions at the source and the online-to-offline transitions detected by the monitoring infrastructure. The noise introduced by this 60 minutes delay did not affect the ranking based on Hamming Difference in presence of the pattern injected during day 1 and day 2, as the target node observation still ranked first among all the observations retrieved.

It does, however, strongly affect the Hamming Distance ranking in presence of patterns such as the one adopted in

[1] IP addresses have been redacted for privacy reasons

[2] Redacted as T.T.T.T

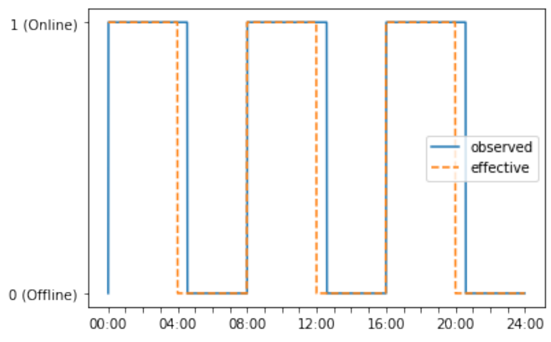

Figure 6: Observed target availability vs. effective target availability - days 1 and 2

address	HD	HD_rank	cumulative
...	...	...	...
i.i.i.i	0.128472	58	3680
j.j.j.j	0.128472	58	3680
T.T.T.T	**0.128472**	**58**	**3680**
k.k.k.k	0.128472	58	3680
l.l.l.l	0.128472	58	3680
...	...	...	...

Table 3: day 3 target node observation ranking, 1.198 unique HD values and 43.458 observations

address	HD	HD_rank	cumulative
...	...	...	...
m.m.m.m	0.124306	54	251
n.n.n.n	0.124306	54	251
T.T.T.T	**0.124306**	**54**	**251**
o.o.o.o	0.125000	55	3480
p.p.p.p	0.125000	55	3480
...	...	...	...

Table 4: day 4 target node observation ranking, 1.204 unique HD values and 43.305 observations

day 3 and day 4, where an online period of 7 hours is followed by an online period of 1 hour. Tables 3 and 4 show that the Hamming Distance of the observed patterns of the target node ranks 58 out of 1.198 distance values in day 3, and 54 out of 1.204 distance values in day 4. In day 3, there are 3.680 node observations within rank 58 out of 43.458 total observations, or 8.4% , while in day 4 there are 251 node observations within rank 55 out of 43.305, or 0.5%. In summary, in presence of the availability pattern injected during day 3 and day 4, the target node observation ranks varies within the 0-10% of the overall node observations.

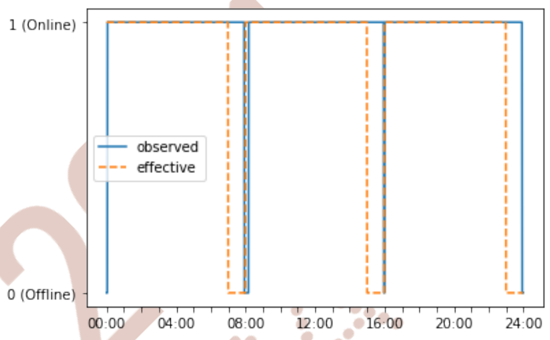

Figure 7: Observed availability vs. target effective availability - days 3 and 4

Figures 3 and 4 highlight why the delay in detecting online-to-offline transitions affects measurements in presence of patterns such as the ones of day 3 and day 4: the length of the offline transition results barely detectable, being observed as only few minutes long compared to the actual 60 minutes of offline effective period. The root cause of this delay can be identified in the standard behaviour of I2P *floodfills* nodes, where a default timeout of 60 minutes is applied before considering a RouterInfo as expired. That is, 60 minutes from the time a node stops advertising its presence via RouterInfo, to the time *floodfill* nodes remove its RouterInfo from the netDB database. The noise introduced by this behaviour can, however, be corrected in several ways: either by introducing discrete filter to the observed measurement, or by modifying the i2pd software in order to reduce the 60 minutes expiration timeout. Tampering with the standard behaviour of the i2pd software is outside the constraints of our experiments, and we have decided to apply a discrete filter as follows.

When an online-to-offline transition is observed, all the previous 60 samples are to be erased and considered as offline. This effectively erases the noise introduced by the 60 minutes expiration timeout, and after having applied the filter as described, the Hamming distance calculated on the target observation ranks first both on day 3 and on day 4. The filtered target node observation ranks first among 43.458 total observations in day 3, and first out of 43.305 observations in day 4. Figure 8 demonstrates how the filtered observation resembles the effective observation.

The same technique, however, cannot be applied on the data collected during the experiments of day 5 and 6, where 11.5 hours of online presence are followed by 0.5 hours of offline absence on a daily basis. Figure 9 shows how the 0.5 hours, or 30 minutes, of offline absence are completely obscured by the default 60 minutes expiration timeout previously described: no online-to-offline transitions are to be observed. The target node

address	HD	HD_rank	cumulative
T.T.T.T	**0.017361**	**1**	**1**
q.q.q.q	0.079167	2	2
r.r.r.r	0.084028	3	4
s.s.s.s	0.084028	3	4
u.u.u.u	0.084722	4	5
...	...	...	...

Table 5: day 3 target node observation ranking (top 5 entries), with filter applied, 1.158 unique HD values and 43.458 observations

address	HD	HD_rank	cumulative
T.T.T.T	**0.047917**	**1**	**1**
v.v.v.v	0.077083	2	2
w.w.w.w	0.079861	3	3
x.x.x.x	0.084722	4	4
y.y.y.y	0.087500	5	5
...	...	...	...

Table 6: day 4 target node observation ranking (top 5 entries), with filter applied, 1.197 unique HD values and 43.305 observations

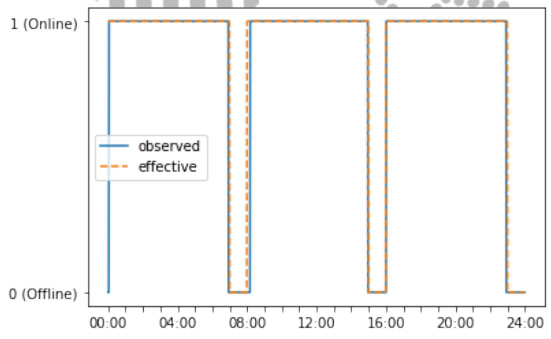

Figure 8: Filtered observed availability vs. effective availability - days 3 and 4

address	HD	HD_rank	cumulative
...	...	...	...
z.z.z.z	0.125	50	3357
a.a.a.a	0.125	50	3357
T.T.T.T	**0.125**	**50**	**3357**
b.b.b.b	0.125	50	3357
c.c.c.c	0.125	50	3357
...	...	...	...

Table 7: day 5 target node observation ranking, 1.198 unique HD values and 42.672 observations

address	HD	HD_rank	cumulative
...	...	...	...
d.d.d.d	0.125	53	3210
e.e.e.e	0.125	53	3210
T.T.T.T	**0.125**	**53**	**3210**
f.f.f.f	0.125	53	3210
g.g.g.g	0.125	53	3210
...	...	...	...

Table 8: day 6 target node observation ranking, 1.201 unique HD values and 41.341 observations

was observed to be online for the totality of the 1440 samples, and the noise introduced in this observation therefore cannot be compensated with any filter. Tables 7 and 8 highlights how the calculated HD distance ranked 50 and 53 respectively in day and day 6, and the node was amongst the top 3.357 out of 42.672 - or 7.8% - in day 5, and amongst the top 3.210 out of 41.341 - or or 7.7% - in day 6. Only a modification of the *flood-fill* I2P node software could compensate this, by reducing the expiration timeout to a period which is smaller than the minimum target offline period, 30 minutes. These modifications are, however, outside of the scope of this experiment, where maintaining the i2pd default because was a constraint.

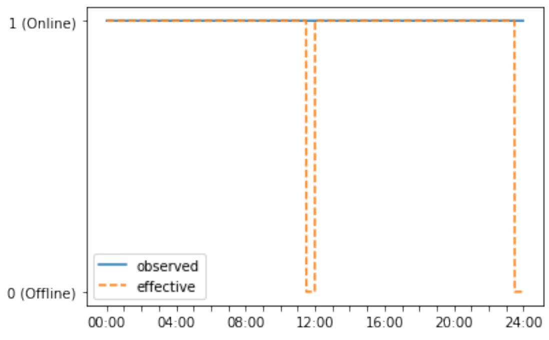

Figure 9: Observed availability vs. target effective availability - days 5 and 6

6. Conclusions

The results demonstrated how the implemented monitoring system has been able to deanonymize a controlled node serving a website via I2P network by inferring its IP address and port number purely based on observations of the node availability over time. More generally, the authors of this research argue that, having enough visibility of an anonymity network and being able to record over time the presence of nodes on the network, a generic approach based on passive observations for the deanonymization of a service delivered via anonymity network is possible.

For the purpose of this research, the standard behaviour of the I2P node software was not altered, and only minor modifications that allowed the sampling of the netDB database were implemented. **Despite our monitoring infrastructure being able to reach a network visibility of only 13%-15% of the estimated I2P network size, in our experiments a target node under observation—where a synthetic online availability pattern was injected—was correctly deanonymized among approximately 40.000 other nodes connected to the network on a daily basis.**

More research is needed for understanding if scenarios with more complex availability patterns can still lead to the deanonymization of a service, either by increasing the network visibility or by relaxing some of the constraints. For instance, behaviour modifications such as reducing the I2P monitoring nodes expiration timeout could overcome the limitations shown in this paper, as well as applying other correlation measures in addition to the Hamming Distance or more advanced filtering mechanisms in place of the one adopted. Future research is also needed in order to generalise this method to other anonymity networks such as Tor, or any other anonymity network where it is possible to observe the availability of its connected nodes.

References

[1] Goldberg, Ian. "Privacy-Enhancing Technologies for the Internet III: Ten Years Later." Digital Privacy, December 22, 2007.

[2] Dalins, Janis, Campbell Wilson, and Mark Carman. "Criminal Motivation on the Dark Web: A Categorisation Model for Law Enforcement." Digital Investigation 24 (March 1, 2018): 62–71.

[3] Kao, Da-Yu, and Shiuh-Jeng Wang. "The IP Address and Time in Cyber-crime Investigation." Policing: An International Journal of Police Strategies & Management 32, no. 2 (May 29, 2009): 194–208.

[4] Dingledine, Roger, Nick Mathewson, and Paul Syverson. "Tor: The Second-Generation Onion Router:" Fort Belvoir, VA: Defense Technical Information Center, January 1, 2004.

[5] Pfitzmann, Andreas, and Marit Kohntopp. "Anonymity, Unobservability, and Pseudonymity – A Proposal for Terminology." In Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2009:1–9, 2001.

[6] Back, Adam, Ulf Möller, and Anton Stiglic. "Traffic Analysis Attacks and Trade-Offs in Anonymity Providing Systems." In Information Hiding, edited by Ira S. Moskowitz, 2137:245–57. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001.

[7] InvisibleNet. "Invisible Internet Project (I2P)," 2003.

[8] Goldschlag, David M., Michael G. Reed, and Paul F. Syverson. "Hiding Routing Information." In Information Hiding, edited by Ross Anderson, 1174:137–50. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996.

[9] Tor Metrics, `https://metrics.torproject.org/`

[10] Tor Metrics, about `https://metrics.torproject.org/about.html`

[11] stats.i2p - The home for netDB statistics, `https://stats.i2p`

[12] Hern, Alex. "US Defence Department Funded Carnegie Mellon Research to Break Tor," February 25, 2016, sec. Technology. `https://bit.ly/us-defence-breaks-tor`

[13] Shirazi, Fatemeh, Milivoj Simeonovski, Muhammad Rizwan Asghar, Michael Backes, and Claudia Diaz. "A Survey on Routing in Anonymous Communication Protocols." ACM Computing Surveys 51, no. 3 (June 12, 2018): 1–39.

[14] Chris Barry GitHub repository for i2spy. `https://github.com/chris-barry/i2spy`

[15] i2pd (I2P Daemon) is a full-featured C++ implementation of I2P client. `https://github.com/PurpleI2P/i2pd`

[16] GitHub repository of the I2P Java implementation. Accessed 20 Feb 2020. `https://github.com/i2p/i2p.i2p`

[17] Docker: Lightweight Linux Containers for Consistent Development and. `Dirk Merkel`

[18] alpine: A minimal Docker image based on Alpine Linux with a complete package index and only 5 MB in size. `https://hub.docker.com/_/alpine`

[19] Kubernetes: Production-Grade Container Orchestration `https://kubernetes.io/`

[20] Cappos, Justin, Matthew Hemmings, Rick McGeer, Albert Rafetseder, and Glenn Ricart. "EdgeNet: A Global Cloud That Spreads by Local Action." In 2018 IEEE/ACM Symposium on Edge Computing (SEC), 359–60. Seattle, WA, USA: IEEE, 2018.

[21] Edge-Net portal `https://edge-net.org/`

[22] Redis `https://github.com/antirez/redis`

[23] Network Time Protocol Version 4: Protocol and Algorithms Specification `https://tools.ietf.org/html/rfc5905`

[24] An Empirical Study of the I2P Anonymity Network and its Censorship Resistance `https://arxiv.org/pdf/1809.09086.pdf`