Evaluating Onion Address Collection Methods

Tobias Höller
Johannes Kepler University
Linz, Austria
tobias.hoeller@ins.jku.at

René Mayrhofer
Johannes Kepler University
Linz, Austria
rm@ins.jku.at

Abstract

This work investigates and evaluates multiple methods of collecting Tor onion addresses to be used in further research. Some of those methods have been used frequently in the past, while others have not been used in research so far. The resulting dataset represents the largest known collection of unique v3 onion addresses with a total of 482.614 unique entries. In order to verify the existence of the collected addresses, several hidden service directory nodes were deployed to harvest blinded public keys. Correlating these keys with the collected onion addresses reveals how many of the discovered onion addresses were active, how much usage they received and what fraction of overall onions they represent. The collected onion addresses were used to unblind more than 25% of the collected blinded public keys which were responsible for 66% of all successful service descriptor downloads.

Keywords

Tor, Onion Service, Hidden Service, Darknet

1 Introduction

Onion services are one of the more controversial features provided by the Tor network. This is mostly due to the fact that they are often associated with illegal marketplaces like Silkroad[1] or the leaking of confidential data stolen from organizations. However, there are also legitimate use cases of onion services. If governments suppress free speech and censor publicly available information, onion services provide dissidents and prosecuted minorities with a way to freely communicate with each other.

This debate has resulted in a lot of research trying to understand what onion services are being used for in practice. A main challenge when doing this kind of research is rooted in the fact that there is no way to enumerate all currently existing onion services. The Tor project does provide an estimate of how many onion services there are—at the time of writing the number is around 800,000 [11]—and how much traffic they receive, but no information on that would enable a user to connect to these services. To connect to an onion service, one needs to know its specific onion address and this address is only known to the creator of an onion service, unless the owner decides to share it with others. This has required researchers to collect large sets of published onion addresses in order to conduct their evaluations.

One consequence of this approach is that research on onion service usage is strongly influenced by the onion addresses the researchers were able to find. If, for example, marketplaces spread their onion addresses far and wide to attract customers, while dissident groups keep their onion addresses within small circles, researchers are far more likely to include the first in their research data. Privacy-preserving applications like Cwtch[2], or Briar[3] are examples of onion services that are unlikely to be published and therefore unlikely to get included into research datasets.

Another aspect to consider is that a small amount of onion addresses is responsible for a large majority of onion service usage, while the majority of onion services is barely used at all [5]. As a consequence, a small dataset containing the most used onion services might represent far more onion service usage than a huge dataset missing those critical services.

This work focuses on the question of how researchers can build collections of published onion addresses to use for further research. We evaluate multiple methods of collecting onion addresses, compare their effectiveness and gather the largest known dataset of v3 onion addresses. Additionally, we combine our addresses with data from a previous experiment designed collect blinded public keys from the hidden service directory [5] to find out if and how often descriptors for the collected onion addresses are being uploaded and downloaded. This enables us to compare the different collection methods not just in regard to the amount of onion addresses found but also in regard to how much onion service usage they are responsible for, a metric providing important context to both previous and future research on onion service usage.

2 Related Work

This section discusses established methods of collecting Tor onion addresses.

2.1 Controlling the Hidden Service Directory

In 2013, Biryukov et al. [3] published a scheme that employed *shadow relays*, that could be injected in the hidden service directory (HSDir) at specific locations. This enabled them to collect the addresses of every onion service active during the day of their data collection. In total, they managed to collect 39,824 unique v2 onion addresses. This constitutes the only known approach capable of enumerating all currently running onion services. After the publication of their results, the Tor network made changes to remove shadow relays from the Tor consensus, making this approach no longer viable.

[1] silkroad6ownowfk.onion (defunct)

[2] https://docs.cwtch.im/

[3] https://briarproject.org/

Free and Open Communications on the Internet 2025(2), 28–36

2.2 Harvesting the Hidden Service Directory

In 2016, Owen and Savage [8] extracted onion addresses from the HSDir by operating multiple nodes within the hidden service directory for a period of six months. On each day, they were able to observe a random subset of all existing onion services and over time they were able to extract almost 80,000 unique v2 onion addresses. Their approach was later used by other researchers as well, with the most successful collection gathering 173.190 unique v2 onion addresses in 2019 [16].

This approach is also no longer viable because v2 onion addresses were deprecated in 2021 [13] and the addresses of the currently used v3 onion services can no longer be harvested via the HSDir [7, 12] because the uploaded descriptors no longer contain their onion address.

2.3 Clearnet Search Engines

Another commonly used method to collect published onion addresses is the use of well known Internet search engines like Google or Bing. If onion addresses have been posted on websites, then a search with the right keywords should be able to turn them up. An important drawback of this and all following approaches is that only *published* onion addresses can be found. This reduces the number of onion addresses that can potentially be found, but it seems reasonable to assume that published onion services are responsible for the majority of onion service usage, so the results should still be valuable for further research.

In 2016, Kang et al. employed this method to discover more than 173,667 unique v2 onion addresses [6], a substantial number considering that at the time there were only about 30,000 v2 onion services deployed per day[11]. Their approach was based on a service called *tor2web*[4], which allows users to access onion services without installing the Tor application on their computer by appending a suffix like `.ly` or `.city` to an onion address. This turns the onion address into to a valid domain name and causes search engines to index the content of these onion services as if they were regular websites.

While search engines can equally be used to detect v3 onion addresses, current research usually combines their output with other information sources. Most of them do not provide a breakdown of how many addresses were contributed by which information source. A notable exception is the work of Pastor-Galindo et al. [10] describing that 759 unique v3 onion addresses were found via the DuckDuckGo search engine (from a total of 80,049). Unfortunately, there are no published results for other search engines, but there are ways to estimate their success from total results. Wang et al. discovered 57,531 unique v3 onion addresses in 2023 by combining the search results of Google and Bing with the results provided by Ahmia[5], the Hidden Wiki[6] and DARKWEBLINKS[7] [17]. Considering that Ahmia currently accounts for about 20,000 addresses it seems unlikely that the number was significantly larger in the past. Applying the same argument the other two sources suggests that they are unlikely to provide more than a few hundred results each. Based on this we estimate that search engines have likely contributed about 30,000 onion addresses to their dataset.

2.4 Crawling Onion Services

Onion services themselves can also be a valuable source of onion addresses. Multiple projects and research works have tried to collect onion addresses by scraping the web content of known onion services. Some of these projects aim to provide the public with a way to search the content of onion service websites like for example Ahmia or OnionLandSearch[8]. Others aim to improve our understanding of how onion services are being used [1, 4, 17]. It should be noted that both Ahmia and OnionLandSearch share their list of known onion addresses publicly, so they are commonly used by other researchers as seed lists for their own crawlers. At the time or writing, the Ahmia list contained 18,069 onion addresses, while OnionLandSearch shared 4,178 onion addresses publicly.

In 2017 Al Nakbi et al. managed to gather more than 250,000 v2 onion addresses by crawling v2 onion services using Ahmia and onion.city[9] as seed sources [1]. More recently, in 2023 Boshmaf et al. ran a crawler to collect onion addresses using OnionDir[10] and Torch[11] as seed lists. They managed to collect roughly 20,000 v3 onion addresses between 2021 and 2022 [4].

2.5 Self-publishing Services

There are also services that encourage users to either share onion addresses specifically or just information more generally. Repositories specialized on providing onion addresses can do regular liveness checks to identify and remove outdated onions, making their content more useful to both users and researchers. The Hidden Wiki and DARKWEBLINKS are two examples for such services that have been used as a source of onion addresses in previous research [17].

More general data sharing services like Pastebin [15], Reddit [15], and Github [10] have also been used to collect onion addresses in the past.

2.6 DNS Leakage

Since onion services use their own top level domain (`.onion`), their names appear like valid hostnames to systems not adhering to RFC 7686 [2]. If a user types an onion address into a regular browser, the onion service will obviously not be loaded, but the browser might still try to contact the hostname by sending a DNS request for the name the onion service in the public DNS system. Even if regular DNS servers are unable to resolve the name, their logs still keep a record of the requested domain name. In this way, users can accidentally leak an onion address if their application is not configured to use Tor, or if there is a typo in their top level domain, for example by typing `.oniion` instead of `.onion`.

Winter et al. obtained 64 hours worth of traffic data for one of the root DNS servers in 2017 and were able to extract 15,471 v2 onion addresses [18]. This approach would also be very interesting for v3 onion services because it would also allow the collection of onion

[4] https://www.tor2web.org
[5] https://www.ahmia.fi
[6] https://thehiddenwiki.org/
[7] https://www.darkweblinks.com
[8] https://onion.live/
[9] www.onion.city (now defunct)
[10] http://oniodtu6xudkiblcijrwwkduu2tdle3rav7nlszrjhrxpjtkg4brmgqd.onion
[11] http://xmh57jrknzkhv6y3ls3ubitzfqnkrwxhopf5aygthi7d6rplyvk3noyd.onion

addresses that were not intentionally published. Unfortunately, there is no data available to indicate how effective this approach would be to gather v3 onion addresses.

3 Ethical Considerations

Since the Tor network is relied upon by its users to protect their privacy, any research and data collection effort has to consider its potential impact on Tor users. Since onion addresses are intended to be shared, knowledge of an onion address enables access to an onion service, but does not compromise the privacy of the onion service operator. Additionally, the changes introduced with v3 onion services ensure that onion addresses can only be discovered, after they have been disclosed by the onion service operator. This leads to the conclusion, that the collection of such onion addresses does not endanger Tor users.

A main concern of this work was to limit the data collection to the minimum. Collection methods like crawling or search engines return significantly more data, than just the onion address. During the evaluation period, every collection method was designed to filter received data with a regular expression that only matches valid v3 onion addresses and only those addresses were stored along with a reference to the collection method used. So for example, our data indicates that an onion address was found on Github, but it contains no further information about the specific file or repository the address was found in.

The most difficult ethical question regarding this research was whether the collected dataset should be made available to other researchers. While this would be beneficial for future research into onion service usage, it might impact onion service operators whose onion service addresses were not widely known before. Such services could receive additional traffic, for example from efforts to crawl onion services [4], which would lead to increased load and more widespread knowledge of the content they offer. A privacy impact for Tor users would occur, if an onion service contains vulnerabilities or misconfigurations that endanger the privacy of onion service users and this onion service is only found by attackers via the published dataset.

This risk must be weighed against the potential benefits of publishing the collected onion addresses. Future research into onion service usage would become more efficient because there is no need for researchers to build their own datasets. Additionally, a shared dataset enables better comparability between the results of different studies.

After discussing this trade-off with Tor's research safety board[12], they advised us that the re-publication of onion addresses without explicit consent from the owners of those onion addresses constitutes a serious risk to Tor users and is therefore considered unethical. Following this recommendation, the collected dataset will not be shared with other researchers.

4 Collection Methods

Most of the presented collection methods have been attempted before, and our results provide an update on how effective they are in 2025. A few methods had to be skipped because they are either no longer possible like for example harvesting the HSDir or require special access to resources like traffic logs of DNS root servers that were not available. We also present several new methods of collecting onion addresses that have to the best of our knowledge not been used in previous research.

4.1 Onion Search Engines and Repositories

Onion search engines and repositories were combined, because most onion search engines also act as repositories by making the set of onion addresses they obtained through crawling available for download. These lists provide easy access to a large amount of current onion addresses. If a search engine does not act as a repository, extracting onion addresses from it becomes more difficult. Ideally, one could construct a search query that matches every indexed page and extract the onion addresses from the search results. If a single query is insufficient, a series of queries can be made with every possible combination of minimum characters. For example, if a search engine allows single character search terms, searches for every ASCII character could be conducted to harvest as many search results as possible.

Both approaches are limited because search engines and repositories often try to keep their results relevant and in order to achieve this with onion services with a large amount of duplicates [10], onion search engines tend to not show multiple search results that lead to the identical page. This means that clones of websites, which make up a significant part of all onion services, are unlikely to be collected in this manner.

There are other limitations that also need to be considered when evaluating onion address extraction from search engines and indices. First, in order to keep their search results current, onion services are removed if they are found to be offline. Due to the volatile nature of the Tor network, this can cause significant changes in the number of results returned by a search engine. To exemplify this issue, during our research the number of onion addresses shared by Ahmia during a week in April 2025 ranged between 18,000 and 22,000.

Another limitation to keep in mind is that the crawling approach used by search engines limits them to onion services that actually provide web services. Onion addresses that run other protocols are usually not indexed and therefore not included in the list of onion addresses provided by these search engines. Finally, many search engines do not want to facilitate criminal activity, so they intentionally remove certain addresses. Ahmia, for example, has a published list of more than 46,000[13] hashed onion addresses excluded from their search index.

While this reduces the value of comparing the amount of onion services extracted in the past with current attempts, there are two relevant questions for future research:

(1) Which search engine is currently best suited to provide a set of onion addresses?
(2) Is there a benefit to combining the results from multiple search engines?

Table 1 provides an overview over the results obtained from several search engines and repositories. It shows that the Ahmia search engine provided the most results, even during a day when relatively few onion addresses were found. The remaining search engines

[12] https://research.torproject.org/safetyboard/

[13] https://ahmia.fi/banned/

Table 1: Overview of different search engines and repositories

Source	ProvidesIndex	Onions
Ahmia[1]	Yes	18,069
FreshOnions[2]	Yes	11,537
OnionSearch[3]	No	11,501
OnionLandSearch[4]	Yes	7,161
Dargle[5]	Yes	7,254
Torch[6]	No	4,190
Total		**36,028**

[1] https://ahmia.fi
[2] freshonifyfe4rmuh6qwpsexfhdrww7wnt5qmkoertwxmcuvm4woo4ad.onion
[3] https://onionsearch.online
[4] http://3bbad7fauom4d6sgppalyqddsqbf5u5p56b5k5uk2zxsy3d6ey2jobad.onion
[5] https://www.dargle.net
[6] http://d6flq6kldlwucbn7q5f6we3377e6k2ro26aozldgj23phd4737rn3hyd.onion

provided fewer results, but they did add a substantial amount of overall address to the result set by almost doubling the amount of onion addresses that would have been found with Ahmia alone.

A special remark should be made in regard to OnionLandSearch. This search engine provides a list of known onion addresses with 4,316 entries, but their search results linked to at least 7,161 unique onion addresses. This might just be an issue on their website, so we decided to include the number of onion addresses we could extract from their search results, rather than their self-published number in this table.

Another issue worth mentioning was encountered with the Torch search engine. This website is only available as an onion service, but there are several links that claim to be the Torch search engine, with most of them being malicious copies trying to lure users on malicious or scam websites. Since we were looking for onion addresses and did not care about their legitimacy we attempted harvesting onion addresses from both the original Torch instance and a malicious clone. This showed that the number of results returned by the clone was extremely limited with less than 150 onion addresses in total. However, the addresses it did return were mostly not included in the search results of the real Torch or other search engines. This highlights that when specifying sources, brand names like Ahmia or Torch should always be used in combination with the address used to access them to avoid confusion between onion sites and their malicious clones.

4.2 Onion Service Crawling

Another established method to gather onion addresses is crawling existing onion services. Since most onion search engines also rely on crawlers to build their databases, this approach can be expected to yield similar results than gathering from onion search engines. Running a crawler directly addresses some of the limitations posed by relying on onion search engines related to offline or unstable onion services as well as non-HTTP services, so it seems reasonable to expect that crawling will gather more onion addresses than harvesting onion search engines.

A notable limitation to consider here is that crawlers require a seed list of onion services. This seed list determines which onion addresses they can find, so different seed lists can lead to different results. Another limitation is caused by the instability of many onion services. If a major forum with many links is temporarily offline while the crawler is running, it might miss a lot of sites and addresses.

This makes it difficult to compare the success rate of different crawlers. For this work, we decided to compare the results obtained crawling ourselves to the results of relying on an onion search engine by running the Ahmia Crawler ourselves and only using it to collect onion addresses instead of indexing the entire content of a website. We used the same seed list used by Ahmia but did not apply Ahmias blacklist in order to properly account for illegal content on onion services as well. Our crawler ran for 20 days in April 2024 crawling more than 2.9 million pages and collecting 48,745 unique v3 onion addresses. 11,809 of which were found to be on Ahmias blacklist, leaving 36,936 that would have been considered for addition to the index. These results confirm that crawling yields significantly more onion addresses than relying on the results provided by onion search engines, even if the same seed lists are being used.

A noteworthy comparison is that other documented crawling efforts [4] have crawled significantly more web pages than us, but still identified fewer onion domains. This is likely due to the regular re-crawling of known onion addresses as well as the use of a different seed list and serves to highlight that crawlers run with different seed lists at different times can produce widely different results.

4.3 Github

Github[14] is a well known public repository for source code, but it is often also used for other forms of structured information like security advisories, reading lists or links to other relevant projects. All of these have a reasonable chance of including references to onion services. Additionally, some software might have static onion addresses in their source code, which could also be discovered by searching Github for onion addresses. This approach was already tried by Galindo et al. [10] and enabled them to find 1,741 unique onion addresses within thirteen days. Opposite to them, we decided to use Github's own code search instead of relying on third parties. Unfortunately, the current code search API provided by Github does not support searching for regular expressions, which is the most reliable way to identify onion addresses. Fortunately, the web functionality of Github does support searching for regular expressions and could be automated with minimal effort. The search results contained 29,973 unique v3 onion addresses.

As this number was significantly larger than what previous research indicated, we also attempted to reproduce the approach taken by Galindo et al and used Grep.app[15] to search for the string `d.onion`. Since the last character of every v3 onion address is a `d`, this search should find every valid v3 onion address along with several false positives, which can be filtered out locally. Using this approach, 37,593 unique v3 onion services could be identified.

This raised the question, if the amount of onion addresses published on Github had significantly increased recently. However, a

[14] https://github.com
[15] https://grep.app

partial review of the repositories containing the found onion addresses confirmed, that they had been sharing roughly consistent numbers of onion addresses over the past three years. A noteworthy observation in this regard is that while searching for an explanation, a Google search pointed us towards a public Github repository containing a text file with more than 12,000 onion addresses that both search attempts had missed. This along with the fact that the search results from Github and Grep.app showed a surprisingly small overlap — only 7,924 onion addresses were found by both searches — leads us to speculate, that both Github and Grep.App have not fully indexed all public repositories for their search function or are excluding files under certain conditions. Therefore, our current results do most likely not represent the number of onion addresses published on Github, but only the number of onion addresses indexed by the two tested search functions. An increase in indexed repositories is one potential explanation for the significant increase in results obtained.

4.4 Search Engines

Using regular search engines to detect onion addresses seems like a fairly straight forward approach at first. Even if search engines like Google, Bing or Duckduckgo are not indexing onion services, they are still very likely to index pages that contain links pointing towards onion services. Even if they cannot follow them themselves, they should still be included in their searchable data, unless it is intentionally removed. It turns out that this approach is not viable, due to the excessive support provided by modern search engines. A simple search for `d.onion`, even if the search term is put in quotes, causes Google to match any sequence that contains the letter d followed by onion somewhere later. For example, a person called David D. Onion gets found due to this search request and even worse, every single mention of *red onions* matches as well, flooding the search results with vegetable related topics and rendering them completely useless for the purpose at hand. Bing shows similar behavior and Duckduckgo just does not return any results at all.

The approach used by Li et al [6] in 2016 which was based on the Tor2Web feature, was also evaluated. While it still works in theory, the number of active Tor2Web instances has reduced significantly in the last years, only *onion.ly* was identified to be still active. Using the search term *site:onion.ly* provides about 2,300 results in Google and 2,230 results in Bing. Duckduckgo does not provide a number of total results when searching. Surprisingly, these results do still contain v2 onion addresses as well, although v2 onion services have been defunct for several years at this point. Extracting these onion addresses is not trivial due to existing protections against crawlers. After some consideration, we decided that a set of at most 2300 v3 onion addresses was not worth the effort.

A more promising approach was identified when looking into other less popular search engines. The Russian search engine Yandex[16] supports `!` as an operator that forces a search term to be included verbatim in a page. This means a Yandex search for `!d.onion` mostly returns websites containing v3 onion addresses. Unfortunately, collecting all of those onion addresses is still difficult because of crawling countermeasures, which cause the collection to be slow and unreliable. Our collection attempt gathered 24,789 unique onion addresses, but it should be emphasized that these results are very likely incomplete. Yandex does not provide an estimate of total search results, making it very difficult to figure out when all search results have been gathered.

Manually reviewing frequently repeating search results also led to the discovery of three new potential sources of onion addresses, that had not been considered before.

4.5 Ransomlook.io

One of the services identified by looking through the Yandex search results, is Ransomlook.io[17]. They maintain a list of ransomware groups and keep track of the onion addresses they use to communicate with their victims or publish their information if they are unwilling to pay the ransom. Unsurprisingly, many of these groups rely on onion services as they would find it difficult to maintain their operations elsewhere.

The idea to utilize threat intel to collect onion addresses has already been used by Pastor-Galindo et al. when they tried to collect onion addresses from six different intelligence feeds [10]. While they managed to obtain 952 onion addresses, just harvesting the current information on ransomlook.io provided us with a list of 1,344 unique onion addresses. This could just be due to an increase in Ransomware activities, but it could also indicate that this service is a better source of threat intelligence.

4.6 Blockchains

Another service identified through the Yandex search was Bitnodes.io[18], a service that provides insight into available Bitcoin nodes. While the authors had been aware that it was possible to operate Bitcoin nodes as onion services, we had previously assumed their numbers to be insignificant. A look into the data provided by Bitnodes.io immediately falsified this assumption.

More than 14,000 nodes or about two thirds of all Bitcoin nodes are operated as onion services in 2025. Assuming that there are about 800,000 onion services in total as estimated by Tor [11], this constitutes 1.75% of all onion services. Since those onion addresses are used within a large peer-to-peer network, it seems reasonable to expect that their share of onion service usage is even higher than their share of onion service addresses.

Since Bitnodes.io also maintains historic data on the Bitcoin Blockchain, it is possible to collect onion addresses which have hosted Bitcoin nodes in the past. This enabled us to gather 93,274 unique v3 onion addresses, making it (to the best of our knowledge) a larger set of v3 onion addresses than anything used in previous research. While onion addresses pointing to Bitcoin nodes might not be the most interesting starting point for research into onion services, it is surprising that such a large set of onion addresses has not been discussed in previous research.

After the successful extraction of onion addresses from the Bitcoin blockchain, other blockchains were also considered as potential sources. Monero for example also allows the deployment of nodes as onion services. We found 159 nodes[19] running as onion services on the Monero blockchain. Unfortunately, there seems to be no

[16] https://yandex.com

[17] https://www.ransomlook.io

[18] https://bitnodes.io/

[19] https://xmr.ditatompel.com/remote-nodes

archive of previous nodes, so no full list could be obtained. Given the much smaller amount of nodes on this blockchain, we assume that the number of missed onion addresses is limited.

While we could not identify any further blockchains containing onion services, the effort spent on this aspect of research was limited, so there is a chance that more onion services could be enumerated by a more dedicated analysis of publicly available blockchains.

4.7 CheckItOnion

The third service identified through the Yandex search is CheckItOnion[20] a service that can be used to check if an onion service is up or not. Once an onion address is entered into the service, it is continuously monitored for its uptime. The service also has a search function that allows users to search the list of monitored onion addresses. This search function can be tricked into revealing all known onion addresses by searching for `.onion`.

Oddly, the search results vary between the onion service version of the page and the one available on the regular Internet, with the first returning massive 32.731 pages of results, while the latter only returns 2,911 pages of results. Harvesting all search results of the onion service version produced a total of 351,910 unique v3 onion addresses. This makes this service the provider of the largest dataset of onion addresses. Previously, the largest datasets of v2 onion addresses contained 250,000 onion addresses, the largest datasets of v3 onion addresses consist of only 80,000 addresses [1, 9, 10].

Due to the main goal of this service being availability checking, the data provided by this service also includes information when the last attempted access to a service was made and if that attempt was successful. This could be used to easily filter this dataset to only contain potentially valid onion addresses and could also mitigate the biggest disadvantage of this dataset, namely the high probability of including mistyped or long defunct onion addresses.

4.8 Shodan

Some well-known websites want to enable users to also access them via onion services. In order to facilitate this, the Tor project has introduced a custom HTTP header `onion-location`[21] that informs a Tor browser that a certain website can also be reached as an onion service. Using an Internet search engine like Shodan[22], it is possible to search for websites which are configured this header and extract onion addresses from them.

Onion addresses obtained in this way are very likely to point to legal websites, since they are also available without Tor and as such their operators are not anonymous. The optional access via Tor onion services is most likely provided to support users with higher privacy demand or users from countries with restricted Internet access that would not be able to access the site otherwise.

Using Shodan to discover websites setting the `onion-location` header resulted in 4.044 websites referring to 3,848 unique v3 onion addresses.

[20] https://checkitonion.online
[21] https://community.torproject.org/onion-services/advanced/onion-location/
[22] https://shodan.io

4.9 Certificate Transparency Logs

Since onion services provide their own encryption layer, most onion services do not use HTTPS, although TLS certificates are available for onion services [14]. The small amount of onion services that uses TLS certificates can be enumerated via certificate transparency logs. They provide a public immutable record of every issued certificate and since certificates contain the domain name they are issued for, onion addresses can be extracted from them.

During our analysis, we managed to extract 489 unique onion addresses from public certificate transparency logs. While this number is small compared to other sources, it might increase in the future if more onion services start using TLS certificates.

4.10 Hunchly

Another potential source of onion addresses are the Dark Web Reports[23] from hunchly[24]. Their data is apparently also based on crawling onion services, but they do not provide a search engine, instead they make their entire collected data available for download as a form of open source intelligence. Analysis of their published data revealed that it only contains 162 unique v3 onion services, making them not very useful as a source of onion addresses.

5 Tracking Onions

Across all presented methods, a grand total of 482,614 unique v3 onion addresses could be identified. This constitutes the largest dataset of onion addresses collected to date. While it was already presented how many onion addresses were collected with each method, there is an open question as to how redundant our findings are. If one method finds the same onion addresses as another method, there is no reason to combine the results of both methods.

Figure 1 shows the relative overlap between the results of the evaluated collection methods. It shows that CheckItOnion for example found more than half (53.23%) of the onion addresses found by Github, while Github only found 9.58% of the onion addresses found by CheckItOnion. This breakdown highlights that onion addresses for blockchain nodes are partially picked up by public search engines and Github, but stay almost completely invisible for other methods. An even stronger pattern emerges with onion addresses collected via Shodan. These addresses are mostly not picked up with any other method.

Since we did not filter our results during collection, there is a high risk that they contain a significant amount of addresses that either never existed because they are the results of typing errors or stopped existing long ago.

In order to weigh our results by significance, we deployed 50 relays within the Tor network designated to join the Tor hidden service directory. While harvesting onion addresses is no longer possible, it is still possible to harvest blinded public keys. Previous research [5, 17] has shown that this method can be used to track how often blinded public keys are uploaded and downloaded. On average our relays observed 0.68% of the HSDir, meaning that an onion service running for every day in 2024 had a chance of 0.68% of selecting one of our relays for the upload of a blinded public key. Since onion services publish their descriptor to 8 HSDir nodes 2

[23] https://www.dailydarkweb.com/
[24] https://hunch.ly/

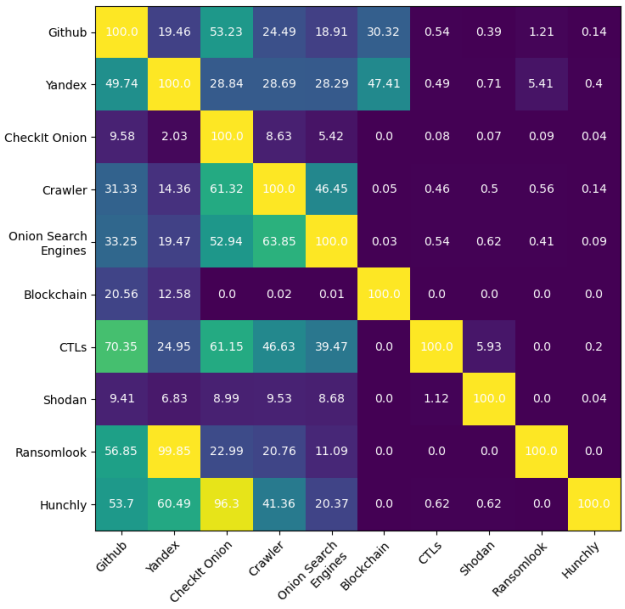

Figure 1: Heatmap visualizing relative overlap between collection methods

`replicas with 4 hsdir_spread`, the probability of an onion service running for 100 days not showing up in our data is only about 3%. While this is a significant limitation considering that onion services are often volatile, the collected data should be sufficient for the purpose at hand.

Our relays[25] have been running from 2024-01-01 until 2024-12-31 harvesting almost 27 million different blinded public keys (BPKs). More than 700 million download requests were processed for these keys, while there were almost 562 million uploads to keep the hidden service descriptors up to date. With our gathered dataset of onion addresses, we can generate potential blinded public keys used by these onion services and then check if those addresses occur within our data. This enables us to estimate how many of our collected onion addresses were active in 2024 and arguably even more interesting, it can be used to quantify how much onion service usage can be attributed to the onion addresses gathered during our experiment. By deriving BPKs from every gathered onion address for every potential time period in 2024, it is possible to quantify the success rate of our onion address collection. Table 2 shows that our dataset was able to correlate 28.15% of all blinded public keys observed. These now unblinded public keys could be used to assign 32.71% of all upload and 46.72% of all download attempts the onion address originally requested by the Tor client. Our observation improves even further, if the dataset is restricted to successful download attempts. In order for a download attempt to be successful, it must be preceded by at least one upload attempt. This restriction makes sense for research trying to understand what onion services are being used for, since failed download attempts do not result in any usage of onion services. Our results show that

[25] https://metrics.torproject.org/rs.html#search/family:008196DC449482C73CFA9712445223917F760921

Table 2: High-level results of correlating collected BPKs with gathered onion addresses

Category	Recorded	Attributed	Share
BPKs	26,973,102	7,591,732	28.15%
Uploads	561,909.886	183,805,449	32.71%
Downloads	703,695,453	328,731,502	46.72%
Successful Downloads	214,184,910	141,752,614	66.18%

our dataset contains onion addresses responsible for 66.18% of all successful descriptor downloads.

In order to accurately evaluate the different data collection methods presented in section 4, their individual contributions to the overall results were also analyzed. Table 3 provides an overview of the relative success achieved with various onion address collection mechanisms. The results show, that the pure amount of onion addresses is no good predictor for the amount of onion usage that can be attributed. Searching Yandex for example yielded less than 25,000 onion addresses, but these could be used to attribute more successful downloads than the more than 350,000 onion addresses collected via CheckItOnion. Despite our issues with searching Github, the addresses that could be gathered were enough to attribute more than 40 % of all successful onion service downloads, making it the most useful source of onion addresses to research onion service usage.

There are several other interesting observations worth noting when analyzing our results. First, the huge set of onion addresses collected from the CheckItOnion service does not contain a significant amount of not-existing onion addresses. More than 99% of the onion addresses collected from this service were also observed by our HSDir nodes. Initially, we speculated that this might be due to the fact that CheckItOnion itself regularly tries to access these addresses and therefore produces download attempts which in turn got logged by our HSDir nodes. But our data disproves that assumption since the amount of uploads registered for the onion addresses from CheckItOnion is not lower than those of other data sources. The number of downloads however, is significantly smaller, indicating that the onion addresses from CheckItOnion are downloaded less often than those of the other data sources.

Another noteworthy observation is that both crawling and onion search engines, the two methods that rely on the content of onion services to find onion addresses, have both performed relatively poor. While they obtained a significant number of onion addresses, their failure rate was the lowest of all search methods, with only 71 % of crawled onion addresses being used in 2024 and only 65 % of onion search results being valid. This is surprising because onion search engines would be expected to keep their results up to date, so why would so many results be missing from our data. A potential explanation for this could be the timing of our research. The collection of onion addresses took place in March and April 2025, while the blinded public keys were collected in 2024. Search engines in their effort to keep their data up-to-date might have provided a significant set of onion addresses that were created in 2025 and are therefore too new to show up in our data. We assume

Table 3: Success rate of different onion address collection mechanisms

Source	Onions	Found Onions	BPKs	Uploads	Downloads	Successful Downloads
Github	63,353	59,282 (93.57%)	1,024,066 (03.80%)	26,224,787 (04.67%)	220,090,472 (31.28%)	94,551,908 (44.14%)
Yandex	24,789	23,384 (94.33%)	382,189 (01.42%)	14,675,276 (02.61%)	154,377,637 (21.94%)	83,178,750 (38.84%)
CheckIt Onion	351,910	350,261 (99.53%)	6,603,930 (24.48%)	164,372,721 (29.25%)	130,447,722 (18.54%)	69,447,116 (32.42%)
Crawler	49,525	35,282 (71.24%)	607,773 (22.25%)	17,563,014 (03.13%)	96,815,673 (13.76%)	55,922,004 (26.11%)
Onion Search E.	36,028	23,515 (65.27%)	356,130 (01.32%)	10,690,719 (01.90%)	55,669,014 (07.91%)	47,025,593 (21.96%)
Blockchains	93,436	76,195 (81.55%)	730,915 (02.71%)	9,426,228 (01.68%)	63,524,868 (09.03%)	27,473,615 (12.83%)
CRTs	489	489 (100.0%)	8,566 (00.03%)	1,302,200 (00.23%)	25,826,707 (03.67%)	24,865,065 (11.61%)
Shodan	3,848	3,654 (94.96%)	64,485 (00.24%)	9,782,910 (01.74%)	8,270,896 (01.18%)	7,935,802 (03.71%)
Ransomlook	1,344	1,211 (90.10%)	16,770 (00.06%)	532,135 (00.09%)	9,508,512 (01.35%)	4,242,398 (01.98%)
Hunchly	162	162 (100.0%)	2,378 (00.01%)	9,367 (00.01%)	233,580 (00.03%)	56,010 (00.03%)
Total	482,614	449,365 (93.11%)	7,591,732 (28.15%)	183,805,449 (32.71%)	328,731,502 (46,72%)	141,752,614 (66,18%)

that both methods would perform better, if the onion address and BPK collection happen in parallel.

This leads to another somewhat surprising observation, the low amount of unseen onion addresses. We expected our data to contain a significant number of onion addresses that do not show up in our data because they are either disabled or were never operated in the first time. Yet, more than 93 % of the onion addresses collected could be attributed to at least one observed BPK. This could indicate that many onion services are unstable but not permanently disabled. Another potential explanation could be the large amount of crawlers searching through running onion services. In future research, we plan to distinguish between BPKs seen in uploads and BPKs seen in downloads to identify the share of active onion addresses.

Finally, it should be noted that the share of onion addresses obtained from different sources can already be used to draw some conclusions about onion service usage. For example, the 12 % of successful downloads attributed to blockchains: While it would be interesting to know why so many Bitcoin nodes are operated in this way and there is a chance that some of them are onion services to hide violations against the terms of service of Internet and electricity providers, their operation is likely not illegal in most countries. A similar argument can be made for onion addresses obtained through certificate transparency logs. Operators who can obtain valid certificates do not require anonymity to shield themselves from legal prosecution. One could reasonably argue that this indicates that almost 25% of onion service usage is not illegal. While this argument requires more detailed analysis that would be beyond the scope of this work, it highlights the impact of an onion address collection method and the importance of understanding the biases introduced by them.

6 Conclusion

We have presented a structured analysis of established onion address collection mechanisms, as well as introduced several new ones that have not been considered in research before. The resulting dataset contains 482,614 unique v3 onion addresses, the largest set of onion addresses ever gathered for research purposes. By attributing almost 30 % of collected BPKs and more than two thirds of onion service usage in 2024, we provide clear indication on how much of the onion service space is represented by our dataset. A surprising result of our evaluation is the low amount of false positives. More than 93 % of all collected onion addresses were responsible for at least one BPK upload or download. Future work will provide a differentiation between onion addresses responsible for uploads only, downloads only, or uploads and downloads.

Evaluating and comparing different data sources also provides insight into the types of research they are best suited for. If researchers are looking for the most commonly accessed onion services, Github appears to be the best source for onion addresses. Researchers interested in why onion services are deployed should rely on the addresses collected by the CheckItOnion service.

Further analysis of the sources and the impact of the collected onion addresses should also be considered in future work. Are the onion addresses collected via a certain method equally contributing to its capability to correlate BPKs, uploads and downloads or is there only a small subset of very active onion addresses that is responsible for the overall results. If only a subset is relevant, identifying this subset would enable more efficient research on onion services.

Even if the addresses collected during this research becomes outdated, documenting the methods used to collect them in the first place, should facilitate the creation of new datasets in the future. This hopefully facilitates future research into onion services and enables researchers to quickly compare new potential data sources to established ones.

Acknowledgments

This work has been carried out within the scope of Digidow, the Christian Doppler Laboratory for Private Digital Authentication in the Physical World. We gratefully acknowledge financial support by the Austrian Federal Ministry of Labour and Economy, the National Foundation for Research, Technology and Development, the Christian Doppler Research Association, 3 Banken IT GmbH, ekey biometric systems GmbH, Kepler Universitätsklinikum GmbH, NXP Semiconductors Austria GmbH & Co KG, and Österreichische Staatsdruckerei GmbH.

References

[1] Mhd Wesam Al Nabki, Eduardo Fidalgo, Enrique Alegre, and Ivan de Paz. 2017. Classifying Illegal Activities on Tor Network Based on Web Textual Contents. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, Mirella Lapata, Phil Blunsom, and Alexander Koller (Eds.). Association for Computational Linguistics, Valencia, Spain, 35–43. https://aclanthology.org/E17-1004/

[2] Jacob Appelbaum and Alec Muffett. 2015. The ".onion" Special-Use Domain Name. RFC 7686. https://doi.org/10.17487/RFC7686

[3] Alex Biryukov, Ivan Pustogarov, and Ralf-Philipp Weinmann. 2013. Trawling for tor hidden services: Detection, measurement, deanonymization. In *2013 IEEE Symposium on Security and Privacy*. IEEE, 80–94.

[4] Yazan Boshmaf, Isuranga Perera, Udesh Kumarasinghe, Sajitha Liyanage, and Husam Al Jawaheri. 2023. Dizzy: Large-Scale Crawling and Analysis of Onion Services. In *Proceedings of the 18th International Conference on Availability, Reliability and Security* (Benevento, Italy) *(ARES '23)*. Association for Computing Machinery, New York, NY, USA, Article 9, 11 pages. https://doi.org/10.1145/3600160.3600167

[5] Tobias Höller, Michael Roland, and René Mayrhofer. 2021. On the state of V3 onion services. In *Proceedings of the ACM SIGCOMM 2021 Workshop on Free and Open Communications on the Internet (FOCI '21)* (Virtual). ACM, 50–56. https://doi.org/10.1145/3473604.3474565

[6] Kang Li, Peipeng Liu, Qingfeng Tan, Jinqiao Shi, Yue Gao, and Xuebin Wang. 2016. Out-of-band discovery and evaluation for tor hidden services. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing* (Pisa, Italy) *(SAC '16)*. Association for Computing Machinery, New York, NY, USA, 2057–2062. https://doi.org/10.1145/2851613.2851798

[7] Nick Mathews. 2018. *Tor 0.3.2.9 is released: We have a new stable series!* Retrieved 2025-03-28 from https://blog.torproject.org/tor-0329-released-we-have-new-stable-series/

[8] Gareth Owen and Nick Savage. 2016. Empirical analysis of tor hidden services. *IET Information Security* 10, 3 (2016), 113–118.

[9] Javier Pastor-Galindo, Félix Gómez Mármol, and Gregorio Martínez Pérez. 2023. On the gathering of Tor onion addresses. *Future Generation Computer Systems* 145 (2023), 12–26. https://doi.org/10.1016/j.future.2023.02.024

[10] Javier Pastor-Galindo, Hông Ân Sandlin, Félix Gómez Mármol, Gérôme Bovet, and Gregorio Martínez Pérez. 2024. A Big Data architecture for early identification and categorization of dark web sites. *Future Generation Computer Systems* 157 (2024), 67–81. https://doi.org/10.1016/j.future.2024.03.025

[11] The Tor Project. 2010. *Tor Metrics*. Tor Tor Project. Retrieved 2025-04-10 from https://metrics.torproject.org/

[12] The Tor Project. 2018. *Tor Rendezvous Specification - Version 3*. Retrieved 2025-03-28 from https://spec.torproject.org/rend-spec/index.html

[13] The Tor Project. 2021. *V2 Onion Services Deprecation*. Retrieved 2025-03-28 from https://support.torproject.org/onionservices/v2-deprecation/

[14] The Tor Project. 2025. *HTTPS for your Onion Service*. Retrieved 2025-04-10 from https://community.torproject.org/onion-services/advanced/https/

[15] Iskander Sanchez-Rola, Davide Balzarotti, and Igor Santos. 2017. The Onions Have Eyes: A Comprehensive Structure and Privacy Analysis of Tor Hidden Services. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE. https://doi.org/10.1145/3038912.3052657

[16] Martin Steinebach, Marcel Schäfer, Alexander Karakuz, Katharina Brandl, and York Yannikos. 2019. Detection and Analysis of Tor Onion Services. In *Proceedings of the 14th International Conference on Availability, Reliability and Security* (Canterbury, CA, United Kingdom) *(ARES '19)*. Association for Computing Machinery, New York, NY, USA, Article 66, 10 pages. https://doi.org/10.1145/3339252.3341486

[17] Chunmian Wang, Junzhou Luo, Zhen Ling, Lan Luo, and Xinwen Fu. 2023. A Comprehensive and Long-term Evaluation of Tor V3 Onion Services. In *IEEE INFOCOM 2023 - IEEE Conference on Computer Communications*. 1–10. https://doi.org/10.1109/INFOCOM53939.2023.10229057

[18] Philipp Winter, Anne Edmundson, Laura M. Roberts, Agnieszka Dutkowska-Żuk, Marshini Chetty, and Nick Feamster. 2018. How Do Tor Users Interact With Onion Services?. In *27th USENIX Security Symposium (USENIX Security 18)*. USENIX Association, Baltimore, MD, 411–428. https://www.usenix.org/conference/usenixsecurity18/presentation/winter