Analyzing inconsistencies in the Tor consensus

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

Michael Roland
roland@ins.jku.at
Johannes Kepler University
Linz, Austria

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

ABSTRACT

Every distributed system needs some way to list its current participants. The Tor networks consensus is one way of tackling this challenge. But creating a shared list of participants and their properties without a central authority is a challenging task, especially if the system is constantly targeted by nation state attackers. This work carefully examines the Tor consensuses created in the last two years, identifies weaknesses that did already impact users, and proposes improvements to strengthen the Tor consensus in the future. Our results show undocumented voting behavior by directory authorities and suspicious groups of relays that try to conceal the fact that they are all operated by the same entity.

CCS CONCEPTS

• **Networks** → Network monitoring; *Network privacy and anonymity*; • **Computing methodologies** → *Distributed algorithms*.

KEYWORDS

Tor, consensus, distributed system

ACM Reference Format:
Tobias Höller, Michael Roland, and René Mayrhofer. 2021. Analyzing inconsistencies in the Tor consensus. In *The 23rd International Conference on Information Integration and Web Intelligence (iiWAS2021), November 29-December 1, 2021, Linz, Austria.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3487664.3487793

1 INTRODUCTION

The Tor project[1] strives to grant users free and open access to the Internet. It avoids both surveillance and censorship by routing traffic via a set of volunteer operated relays that make up the Tor network. Currently, the network is comprised of more than 7000 relays that allow users to anonymize their network traffic via onion routing. Thanks to projects like the Tor browser that makes anonymous browsing easy for most users, Tor has become the unofficial standard for anonymous online communication.

A fundamental requirement for the privacy guarantees provided by the Tor network is that it consists of a large amount of relays controlled by independent operators. This is achieved by allowing and encouraging both private individuals and organizations to operate Tor relays within their networks to share some of their internet bandwidth.

This results in a common problem for distributed systems: How can a user of the system find out about other members? A popular example for this scenario would be the BitTorrent [2] protocol that enables decentralized file distribution. Clients interested in downloading a file need a reliable way of finding other members in the network that have the file available for download. This is achieved by trackers that keep track of users who have already downloaded the file and provide new downloads with potential peers that they could get the file from. This approach works because every file distributed via BitTorrent is completely independent of any other file also using the distributed BitTorrent system.

Another example for this problem is encountered when building blockchain applications like Bitcoin [12]. Before blocks can be verified and appended to a blockchain, the blocks need to be distributed to the Bitcoin participants commonly known as *miners*. This distribution of both transactions and new blocks requires a deterministic way of disseminating information across all current Bitcoin participants. Bitcoin relies on a fairly simple broadcasting approach that requires every node to know at least one other node. If a new broadcast message (either a new transaction or a new block) is received by one node, it will be forwarded to all other nodes within the network. While this approach is obviously not the most efficient or reliable one, the strong cryptographic properties of the blockchain are sufficient to ensure the integrity of the stored information, even if some messages are not received by all participants.

The Tor project has to solve a similar problem, but with some additional complexities. First, it is assumed that any single node could be controlled by an attacker, so a simple broadcasting system would risk leaving individual users without valid information. This could lure users into using untrusted relays and consequently compromise the privacy they were trying to protect by using the Tor network. The strategy employed by BitTorrent is also insufficient, because more advanced applications like onion services depend on all Tor users having the same knowledge about the network. Tor's solution to this challenge is the Tor *consensus*, a single document published hourly that contains all currently available Tor relays along with their most important properties. Section 2 will discuss the procedure of creating a Tor consensus in more detail.

While it is fairly easy to deploy a Tor relay that gets accepted into the consensus, even experienced operators can have trouble understanding the attributes they get assigned in the Tor consensus. This is caused by both Tor's inherent privacy focus and the constant back and forth between malicious actors trying to attack the Tor network and operators taking defensive measures to deflect detected attacks. One such scenario that triggered this research occurred in March

[1] https://www.torproject.org/

iiWAS2021, November 29-December 1, 2021, Linz, Austria
© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-9556-4/21/11…$15.00
https://doi.org/10.1145/3487664.3487793

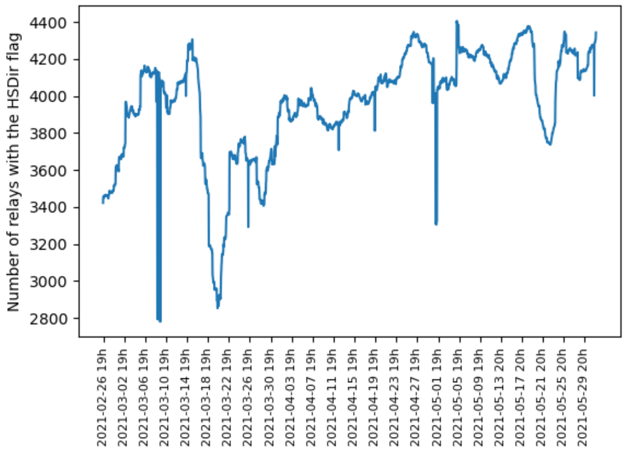

Figure 1: Number of HSDir relays in the Tor network

2021, where without obvious reason, all the Tor relays we were operating at the time for another research project [6] were suddenly no longer considered part of the hidden service directory. When we noticed this, we first went to see if there had been any changes to the requirements for Tor relays, which turned out not to be the issue. Then we checked if other relays were having similar issues. Figure 1 shows that around March 15th, 2021, the number of relays allowed to join the hidden service directory started plummeting from more than 4200 to less than 3000. This means that more than 25 % of all HSDir relays were affected by this event which lasted for several days until the number of relays stabilized again at around 4000. More generally, we could not help but notice that the number of relays trusted with this specific task was fluctuating more than we would have expected. Ideally, the number should go up if new relays are joining the network and go down if relays are leaving the network. Apart from that, there is a possibility that relays have to be actively removed from the network if they are found to be malicious, but what we see here is that relays are only removed from the hidden service directory for short periods of time.

A similar graph to Figure 1 is also provided by the Tor project itself[2] so our observations are also publicly documented. The sharp spikes in our graph that are missing on the official one can be explained by the fact that the Tor project aggregates the numbers per day. This removes sharp spikes like the one on March 9th, where the drop only lasted for a single hour. So those are not errors in our data, they are simply the result of plotting with higher accuracy. We feel this is justified since the consensus with the reduced amount of trusted relays was still valid for one hour during the day, so it should not be ignored.

The Tor network is very transparent, so we do have extensive documentation and discussions on how the Tor network should function in theory. The goal of this research is not to verify or improve the current consensus finding procedure, but to analyze how that procedure works out in practice, why the number of relays trusted with certain tasks fluctuates so much, and if there are any potential improvements that should be considered.

2 THE TOR CONSENSUS

For clients to utilize the Tor network, they need extensive knowledge about the currently available Tor relays. This includes their network addresses, bandwidth, uptime, cryptographic information and more. However, downloading or updating all this information for a large set of Tor relays consumes too much bandwidth to be useful for average clients. To address this issue, Tor publishes different documents to describe the current state of the Tor network. Tor's directory specification [11] documents which information is to be inserted in the different documents. The most important of these documents is the Tor consensus, which contains all currently running Tor relays, along with their fingerprints, attributes and capabilities. Every Tor client needs a valid Tor consensus to select Tor relays to be used for a new connection. Extended data about a relay, like its cryptographic keys are stored in separate descriptor files (e.g. the server descriptor and the micro descriptor) which are downloaded only if needed.

This makes the Tor consensus an integral component of the Tor network. If it is missing, clients are unable to use the Tor network and if it is forged, clients can be tricked into using malicious relays which identify users who try to stay anonymous. Since there is no single authority sufficiently trusted by the entire Tor network to create and publish the Tor consensus, this task has been distributed across several directory authorities. Every new Tor relay announces itself to all currently running directory authorities and has to wait until they include it in the Tor consensus before it will receive any client traffic. At the moment there are nine directory authorities (*moria1*, *bastet*, *longclaw*, *Faravahar*, *dizum*, *gabelmoo*, *tor26*, *dannenberg*, *maatuska*) running in 6 different countries in North America and Europe.

Every directory authority maintains an independent view on the Tor network and publishes its perspective hourly in a network status vote. The consensus is also created hourly by every directory authority but without regard for their personal view of the network. Instead, they collect the network status votes from all available directory authorities (including their own) and include everything in the consensus that is included in a majority of votes. So a relay is only included in the consensus, if more than 50 % of the votes include the relay. This also applies to properties of a relay, meaning a relay is only considered fast if more than 50 % of the votes believe it to be fast. If everything works as intended, all nine directory authorities have access to all nine votes and produce the same consensus, which they sign digitally before publishing it. By exchanging signatures, each directory authority ends up with a consensus document that has been signed by all other directory authorities. Note that while an ideal consensus is built from nine votes and has nine signatures, a valid consensus only requires the signatures of a majority of voting directory authorities.

2.1 Flags

The Tor network describes the properties of Tor relays with a series of flags that are assigned by the directory authorities if the relays meet the necessary criteria. The following list provides a selection of the more important flags currently present in the Tor consensus:

[2] https://metrics.torproject.org/relayflags.html

- *Valid:* Assigned if the version of Tor run by the relay is not known to be broken. Invalid relays are not included in the consensus.
- *Running:* Assigned to all running relays. Requires the directory authority to be able to connect to the relay. Relays that are not running are not included in the consensus.
- *V2Dir:* Assigned if the relay supports the V2 directory protocol. Unless actively disabled, all current Tor versions obtain this flag.
- *Fast:* Assigned if the relay is suited for high bandwidth (≥ 105 KB/s) connections.
- *Stable:* Assigned if the relay is suited for long-lived connections. Requires the relay to have a mean-time-between-failure of more than seven days.
- *HSDir:* The relay is part of the hidden service directory. Assigned only if the node is stable, fast and has been up for more than 96 hours.
- *Guard:* Assigned if a relay is suited to be the first node of a Tor connection. Requires a relay to be fast, stable, be a V2Dir, have an at least median uptime, be at least a few weeks old, and have a bandwidth of more than 2 MB/s.
- *BadExit*: Assigned if a directory authority believes that an exit relay should not be used by clients. Unusual because there is no specification on how that assumption should be built, the only example given is using an internet provider that is known to block/censor traffic. In practice, the assignment process for this flag is semi-manual.

2.2 Tor Bandwidth Authorities

While it is easy for a directory authority to keep track of the uptime of relays because relays have to upload new descriptors regularly, measuring their bandwidth is more challenging yet still important.

Relay operators usually think of bandwidth in terms of what they pay their internet provider for and make a fraction of that bandwidth available to the Tor network. But the technically available bandwidth does not always correspond with what clients pay for, leading to Tor relays advertising more bandwidth than they can actually handle. Even worse, malicious relays can advertise huge amounts of bandwidth that they could never handle, just to cause the Tor network to send them lots of traffic that they will drop. In both cases the user experience for all Tor users suffers from incorrect bandwidth information.

To address this issue, Tor uses several bandwidth authorities [7] which are responsible for measuring the available bandwidth of relays. Since the results of these bandwidth measurements must be incorporated into the votes for the consensus, only directory authorities can be bandwidth authorities. This means that for every consensus vote, some authorities make bandwidth decisions based on advertised bandwidth, while others decide based on their measurements. To prevent malicious relays from only responding to measurement traffic, these measurements must also take place via the Tor network in order to appear just like regular traffic. This means that every bandwidth measurement is going via several nodes, making it hard to tell for certain if the measured relay really was the bottleneck during the measurement. Currently, Tor has two different algorithms for measuring relay bandwidth in use (torflow[3] and sbws[4]), resulting in three different ways how a directory authority can determine bandwidth information about relays.

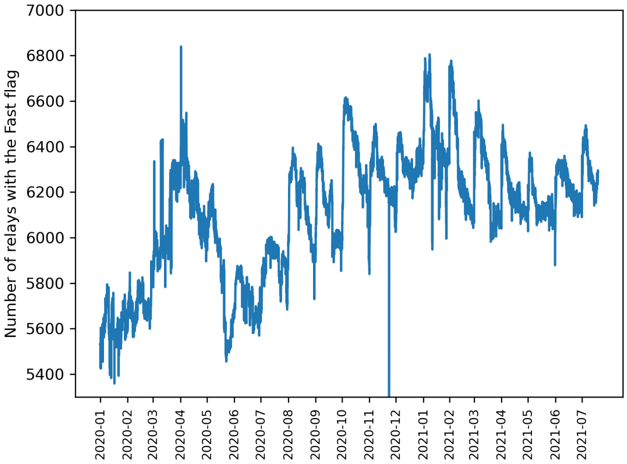

Figure 2: Number of Fast relays in the consensus

3 ANALYSIS

During our analysis of the Tor consensus we confirmed that many things work exactly as expected. In this work we will not discuss any confirmatory results and instead focus on unexpected behaviors and intriguing observations that we encountered during our analysis.

3.1 Data Sources

To analyze and evaluate the decisions made by directory authorities in the past, one needs access to as much information as possible about the Tor network. Thankfully, the Tor team archives [8, 9] all documents made available to clients since 2007, so we do not have to worry about data collection for our research. Instead, we can just use the official data archive, meaning that all of our results and graphs reflect the state of the Tor network according to their own archives.

We have access to all published consensus documents, the votes that were used to create the consensus, as well as the server descriptors that provide extended information about every relay. This data enables us to find out which and how many directory authorities supported every single decision that went into the Tor consensus, and provides the foundation for all the results presented in this section.

3.2 Fast relays

While our interest in this research was triggered by the variations in the assignment of the HSDir flag, we start our analysis with the Fast flag, because it is a prerequisite for the HSDir flag and due to the three different methods of determining bandwidth, we believed it to be the most likely reason for relays temporarily losing the HSDir flag.

[3] https://gitweb.torproject.org/torflow.git/tree/NetworkScanners/BwAuthority/
[4] https://gitlab.torproject.org/tpo/network-health/sbws

According to Figure 2, this theory appears to be incorrect, because the amount of relays granted the Fast flag does not fluctuate nearly as much as the amount of relays with the HSDir flag. This also matched our own experience since our relays had retained their Fast flags during the period where they were not granted the HSDir flag.

At this point it is important to remember that obtaining the Fast flag only means that more than half of the directory authorities believed the relay to be fast. Directory authorities which do not believe a relay to be fast (or stable for that matter) will never consider granting the HSDir flag. For example, if a relay gets 5/9 votes for the Fast and Stable flags, it will obtain both flags, but if just a single one of those 5 authorities does not believe the relay to be up for more than 96 hours, it will not obtain the HSDir flag.

To visualize this aspect, we parsed the archived votes of all directory authorities and evaluated for every relay in the consensus how many votes for the Fast flag it received. The results of this analysis are visualized in Figure 3 and show that most relays received the Fast flag with 100 % of the votes. It also confirms several of the voting patterns we expected to see based on the current state of bandwidth measurement strategies. There are almost no relays that receive one or two votes for the Fast flag, but there is a noticeable chunk that receives three votes. This is caused by the directory authorities that are not bandwidth authorities, as they have to believe the data reported by the relays themselves.

At first, it seems like these are relays that advertise more bandwidth than they can actually provide. However, that is not exactly what the measurement tells us because bandwidth tests are obviously taking place in parallel with ordinary operation. If a relay is already handling one other connection during a bandwidth test, it will split the available bandwidth between both connections, meaning that the measured bandwidth is much lower than what was actually made available to the Tor network. A critical setting in this regard is the *MaxBandwidthBurst* configuration option, that tells Tor what amount of bandwidth it is allowed to consume at most. When we deployed relays with a bandwidth and bandwidth burst limit of 105 KB/s, they were never found to be Fast despite all of them fully providing their advertised bandwidth. Only after increasing the burst rate to several times the bandwidth rate, relays started to measure as Fast. It may therefore be the case that relays advertising enough bandwidth for the Fast flag do not receive this flag because they don't have the needed burst capabilities to be measured as such.

Also worth mentioning is the fact that the relative share of relays that are considered Fast by all directory authorities is increasing, so it seems like the Tor consensus is actually getting more stable in that regard. This improvement is likely caused by the fact that internet bandwidth is increasing globally and most web applications and services expect users to have more than 105 KB/s of bandwidth available. Raising the requirements for the Fast flag might be a good idea to improve user experience when using the Tor network.

Another interesting observation we made when analyzing the voting behavior on the Fast flag was that the number of voting relays is regularly lower than nine. This includes February and March 2021, where the Tor consensus regularly contained only 7 or 8 different votes. In theory, the Tor consensus should be very resilient to the failure of individual directory authorities, so there has to be another factor causing the high volatility of the HSDir flag.

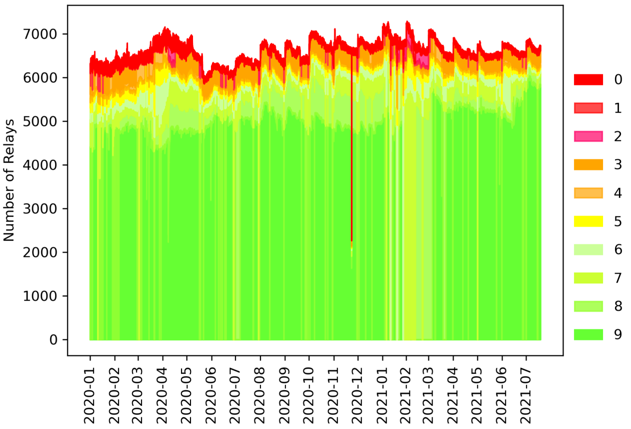

Figure 3: How many votes for the Fast flag did relays receive

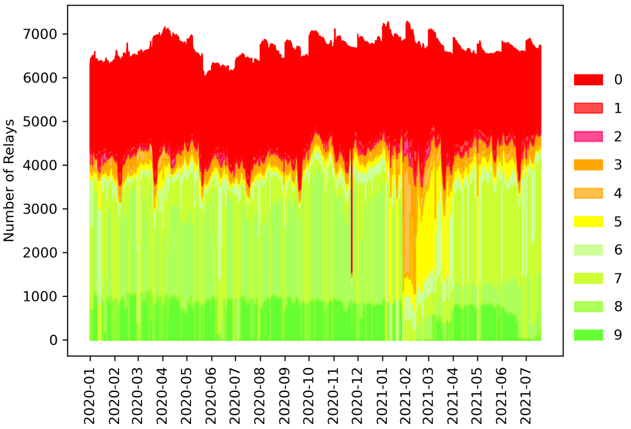

Figure 4: How many votes for the HSDir flag did relays receive

3.3 HSDir Relays

Figure 4 visualizes the number of votes for the HSDir flag received by relays in the consensus. It clearly shows that the number of relays that receive the HSDir flag from all directory authorities is less than 1000 meaning that more than 75 % of all HSDir relays in the consensus are relying on only 6 or 7 votes instead of 9. Secondly, we can clearly see a negative spike around February 2021, where the number of relays with 4 and 5 votes suddenly spikes. Since this is the time when some directory authorities stopped voting, this confirms that the non-voting directory authorities were the ones relays previously relied upon to obtain the HSDir flag.

But the real question to ask at this point is why the level of disagreement between the votes of the directory authorities is so much larger for the HSDir flag than it is for other flags like Fast. For that purpose, we define a new metric: The amount of dissenting votes. A dissenting vote happens when a directory authority granted or

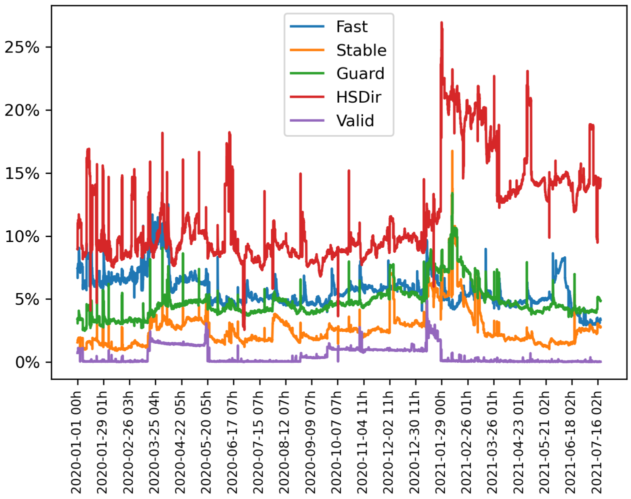

Figure 5: Dissenting votes per flag

withheld a flag in a vote, which ended up being granted by the consensus.

Figure 5 shows the relative amount of dissenting votes for different flags. Unsurprisingly, it confirms that the number of dissenting votes is highest for the HSDir flag, reaching up to 25 % of the overall votes. Considering that the maximum amount of possible dissenting votes is limited at 44.5 % for nine voting directory authorities, this level of dissent is reason for concern. The fact that both Fast and Stable have significantly lower levels of dissent than HSDir seems to imply that some directory authorities have trouble confirming the 96 hour uptime of relays. Uptime is tracked but not published by directory authorities, so there is no easy way to confirm this assumption. However, there is an easy way to disprove it by checking if a directory authority considered a relay Running for the last 96 hours. Checking the archived votes reveals that some directory authorities do not grant the HSDir flag to relays, even if they believe them to be Fast, Stable and Running for more than 96 hours.

Figure 6 shows how many dissenting votes for the HSDir flag were issued by each of the directory authorities. Note that the directory authority moria1 has a significantly different voting behavior for this flag. While other directory authorities tend to have a very low number of dissenting votes with occasional spikes that can be explained by temporary issues when measuring uptime or bandwidth, moria1 constantly disagrees on at least 3000 votes. This nicely aligns with the previous observation from Figure 4 which shows that only a small amount of relays manages to obtain 100 % of votes.

This behavior seems to be intentional, as the operator of moria1 is one of the original developers of the Tor software and is known to use his directory authority to test future changes and improvements to the Tor network [5]. While we do not believe that this is a very good argument as there are ways to test potential future flag requirements without actively introducing dissent in the current consensus, a single directory authority not following the directory specification does not explain the observed fluctuations in the number of HSDir relays.

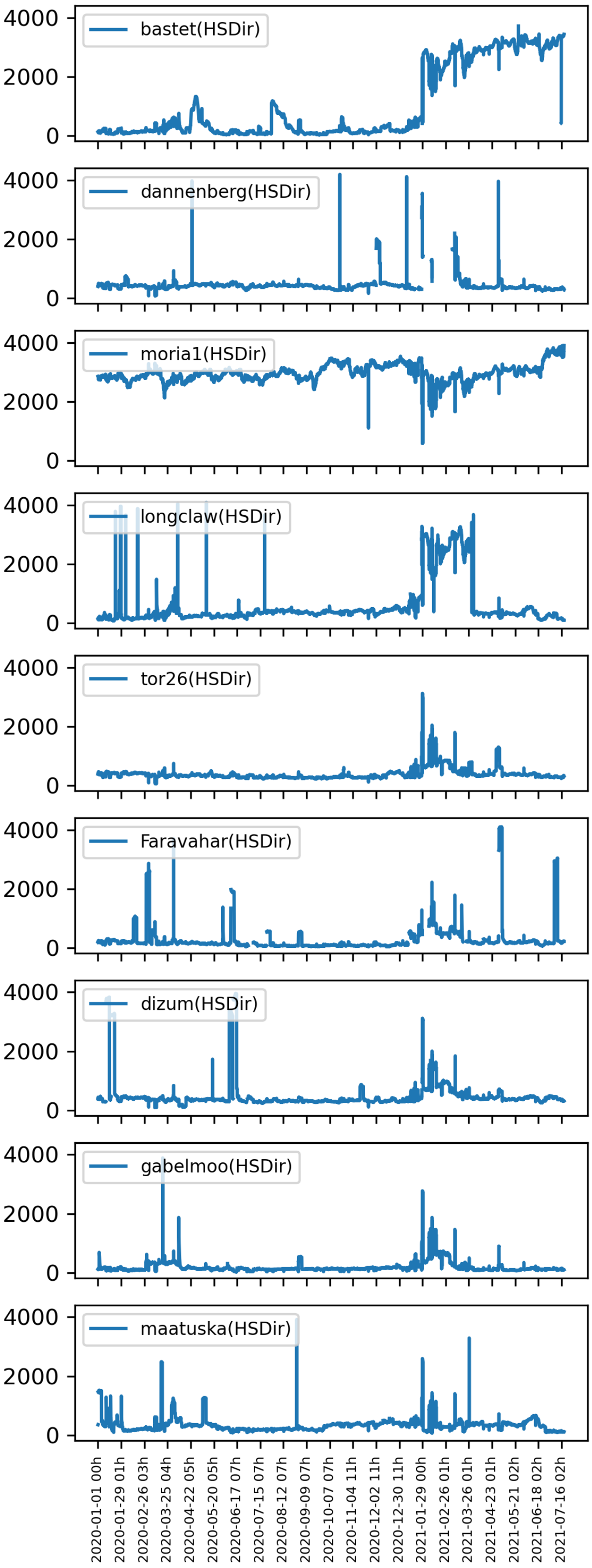

Figure 6: Dissenting HSDir votes per relay

To answer this question, special attention should be paid to the events around January 28th, 2021 when the directory authorities bastet and longclaw suddenly changed their voting behavior to align with moria1. According to the relay operator mailing list [3] a denial of service attack against directory authorities was detected on that day that forced several directory authorities to go offline. Based on this observation, we theorize that in response to this attack the Tor team developed a quick fix on top of the development branch that moria1 was using and made that available to other directory authorities as well to stabilize the network. That would make the changed voting behavior for HSDir flags an unintended side effect. Unfortunately, there is no way to confirm this theory because Tor relays only publish their version string without any further indication of the actual source code they are running. Both before and after January 28th, 2021 moria1, bastet and longclaw published the version string *0.4.6.0-alpha-dev* indicating that their Tor binary was compiled off a development branch. The fact that these directory authorities changed their voting behavior without changing their version string clearly illustrates that current version information provides little insight into what code is actually being run by a relay.

What we have been unable to confirm is whether this dissenting voting behavior is intentional or not. Longclaw returned to the officially specified voting behavior at the end of March 2021 but bastet did not, although their inconsistent voting behavior was reported [4], so the Tor project must be aware of it. Interestingly, when longclaw reverted to voting according to the directory specification, their version string did change to *0.4.5.7*. So they moved from a development build to an older official Tor release. This leaves us wondering if two authorities voting based on different criteria than the others provides any benefits to the Tor network that justify the dissent they are causing.

Ultimately, the high fluctuations in the hidden service directory were caused by a mixture of several issues. First the changed voting behavior of three directory authorities reduced the amount of obtainable votes to six. If any of the remaining six relays went offline – which tends to happen during ongoing DOS attacks – relays needed to obtain five out of five available votes. So any individual measurement failure regarding either bandwidth or uptime led to a withdrawn HSDir flag.

3.4 Other voting inconsistencies

After noticing the different voting behaviors for the HSDir flag, we obviously asked ourselves if the same issue also applies to other flags. For that purpose we ran the same evaluation for all the other flags that can be assigned to relays and found two more hints towards inconsistent voting criteria.

The first again seems to be tied to moria1 and applies to the flags Valid, Exit and V2Dir. Figure 7 only shows the dissent for the Valid flag because the graphs for the other flags look exactly the same, which leads us to believe that the dissent for all three flags was caused by a common issue. Our data shows that the dissenting votes from moria1 started on September 26th, 2020 and continued until January 28th, 2021. The relays bastet and longclaw adopted the voting behavior from moria1 on January 12th, 2021. At this

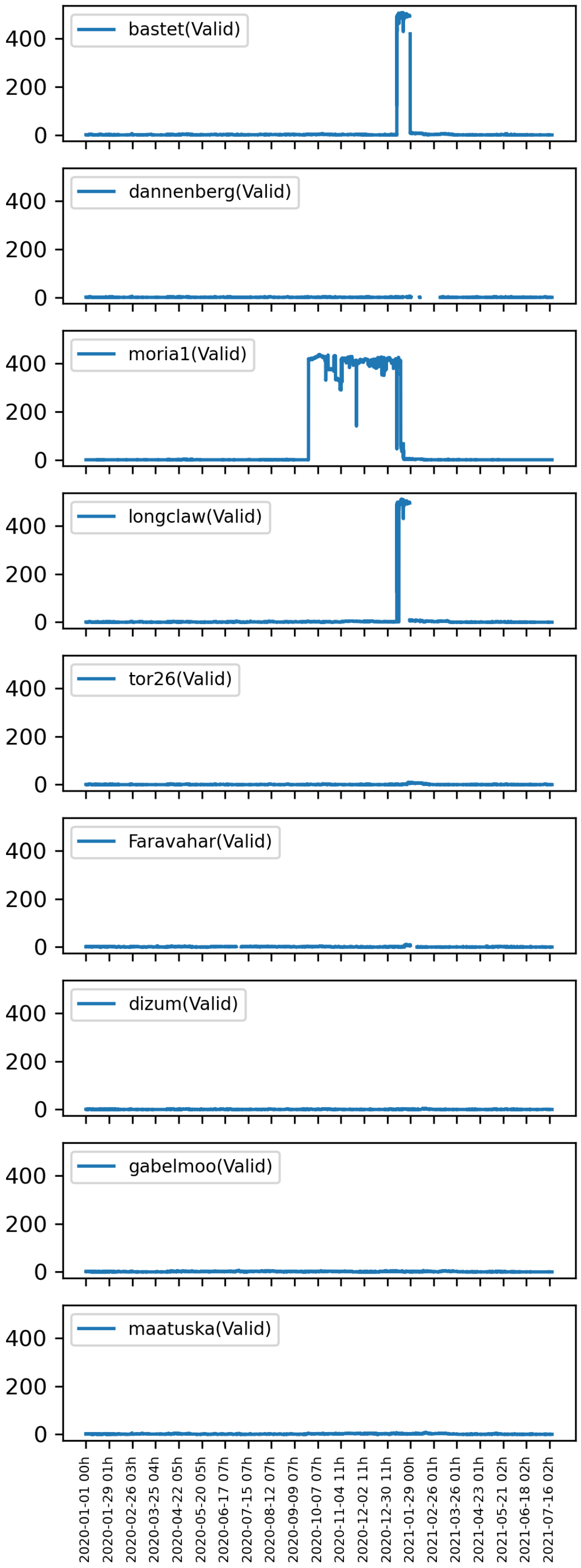

Figure 7: Dissenting Valid votes per relay

point we have to consider the adoption of unspecified voting behavior from moria1 by bastet and longclaw a pattern. Ironically, the change that ended this inconsistent voting pattern by all three directory authorities also caused bastet and longclaw to begin voting against specification for the HSDir flag. So in a way we traded one inconsistency for another.

The second inconsistency we found concerns the BadExit flag. This flag is a little different from the previous ones, because there are no clear requirements as to what a relay must do to earn this flag. According to the specification, this flag should be given to exits that are believed to be useless as an exit node. This would be the case if an ISP censors outgoing traffic or a firewall is too restrictive and prevents Tor users from actually reaching the resources they are interested in via this exit relay. Since there are no clear criteria defined, a group within the Tor project tries to monitor the network for bad exits and flags them as such. While they certainly are utilizing automation for this task, a non-negligible part of their work relies on Tor users reporting relays that do not work as expected.

Considering the fact that bad exits are actively monitored, we were quite surprised to see that the dissent on bad exits in Figure 8 also shows two clearly unique patterns. Bastet and dizum hold a reproducible minority opinion regarding the BadExit flag. Maatuska started supporting them in September 2020, leaving us again with three directory authorities that seem to consistently vote differently from the other directory authorities. Since there are no requirements specified for the BadExit flag, we are unable to find out if this behavior is in fact caused by two different sets of criteria for the flag, or if there are two different measurement methods, or if this actually valid because some relays only work from the perspective of certain directory authorities. Without internal knowledge about the working of the Tor bad-relays team, it is impossible to investigate this phenomenon any further.

3.5 Monthly relay spikes

The final observation we would like to present in this work regards the composition of the Tor consensus. Attentive readers may have already noticed in Figures 3 and 4 that the total number of relays that are being voted on follows a specific pattern that spikes at a certain point in time and then decreases for a while before spiking again. Figure 9 highlights this behavior more clearly and shows these spikes reliably occur on the first of every month since July 2020. To be more precise they all join the Tor network within the first minute of the first day of a new month. There is no clear pattern as to when they leave, but it seems like they randomly drop out of the network over time. This behavior was independently noticed by the Tor project[5], but so far they have no explanations as to why it happens. The findings presented in this chapter were of course made available to the Tor project prior to publication.

The first question we asked was if those spikes were caused by new relays joining the network on a monthly basis or old relays rejoining. By analyzing the archived consensus information we were able to identify 90 relays that had rejoined the network at the beginning of every month since July 2020. Furthermore, we found 230 relays who contributed to at least 10 of the 13 recorded monthly

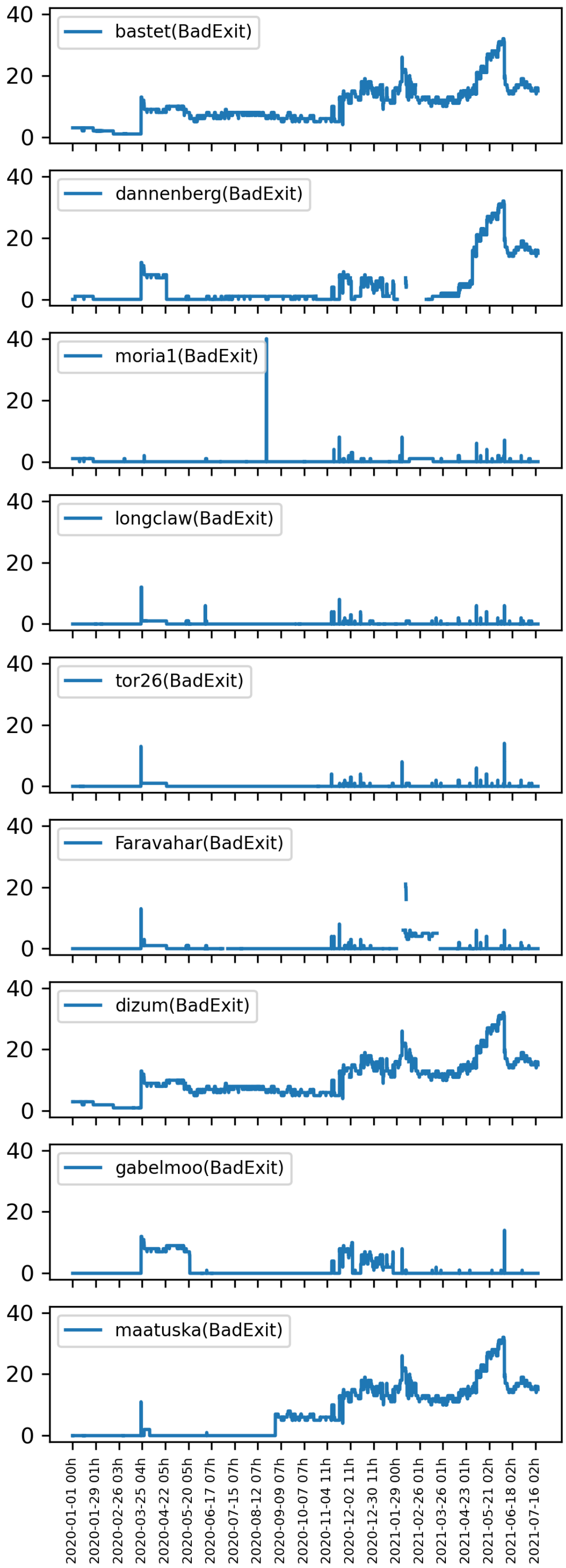

Figure 8: Dissenting BadExit votes per relay

[5] https://gitlab.torproject.org/tpo/network-health/team/-/issues/76

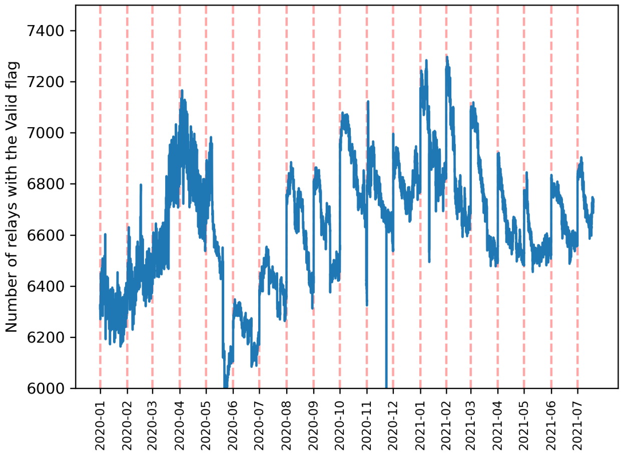

Figure 9: Highlight monthly spikes in number of valid relays

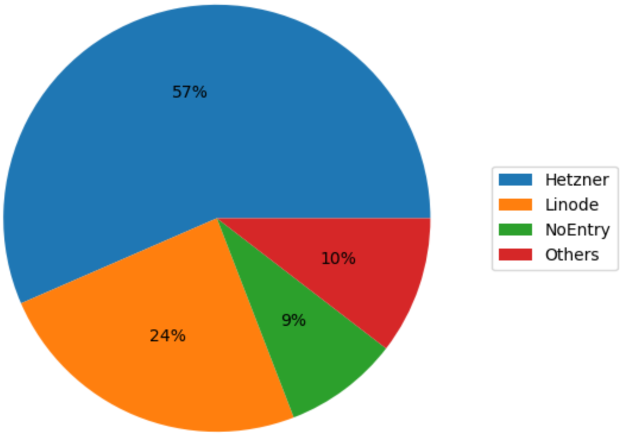

Figure 10: Provider assignment based on hostname for 230 relays that spiked at least 10 of 13 times

relay spikes. So we can confirm that these relay spikes are caused by relays regularly joining and leaving the Tor network.

The regular timing supports the theory that this pattern is produced by a common misconfiguration shared by all those relays. One of Tor's configuration options enables a relay to limit the amount of bandwidth used during a given time interval (*AccountingMax*). This is very useful for users with a strict limit on how much bandwidth they are allowed to consume and could be a potential explanation. If a relay is configured to use a limited amount of bandwidth per month, the observed pattern of relays joining the network at the beginning of a new month and leaving randomly when they run out of bandwidth makes sense. The only problem with this theory is that Tor should not behave like this when this option is enabled. According to the documentation[6], a relay that runs out of bandwidth hibernates until a random time within the next time period to avoid all relays starting at the same time. Unless there is a bug affecting several Tor implementations, this theory does not explain the regular monthly spikes, but it may very well explain why these relays leave the Tor network after a random period of time.

To find out if the relays responsible for this phenomenon have anything in common that might shed light on the subject, we extracted information about them from the consensus documents and relay descriptors published at the beginning of every month. Apart from the unique fingerprint that we used to identify rejoining relays, the Tor consensus reveals the IP address and the Tor version running on the relay. Additionally, the consensus provides the digest needed to query server descriptors with more information about a relay, like its uptime, family or contact information. Additionally, we used reverse DNS lookups to assign hostnames to the IP addresses of the relays. Unfortunately, there were no obvious commonalities between the different relays. The only thing of interest is that reverse DNS responses tie a majority of relays back to very few large cloud hosters. Figure 10 shows that most of the relays that spiked more than 10 times were hosted at the German Hetzner Online GmbH[7] which is one of the largest operators of Tor relays. The other hoster, Linode, LLC[8], is also responsible for a significant amount of Tor relays. While it is common for relays to be run at cloud hosters, these two cloud hosters contribute to these relay spikes far more than they contribute to the overall amount of relays. Other cloud hosters that are used to operate lots of relays like OVH do not show up in our data at all, so the issue seems to be related to these hosters in some way. This argument gets even stronger when we compare the total number of relays operated at those providers to the number of relays with monthly reappearances. Hetzner operates 440 relays, of which 146 (33 %) are contributing to the monthly relay spikes. For Linode the relation is even worse with 216 (54 %) relays total of which 118 contribute to relay spikes. This leads to the conclusion that there is either a widely distributed Tor setup that uses an accounting limit and forces a reboot of the Tor process at the beginning of every month or there is one actor running all those relays on different cloud providers which happens to have an accounting limit and a monthly reboot policy in place.

The tutorials for running Tor relays on both cloud providers [13, 14] do include an accounting limit, but say nothing about monthly reboots and we could not find any public resources that would explain a large number of users setting the same monthly reboot policy. On the other hand however, we detected a weird pattern in when relays that contribute monthly spikes first joined the network. A vast majority joined between April and June 2020, and they did so in ordered time intervals. For example, between April 29th and May 10th, 39 relays that contribute to relay spikes were deployed at Hetzner. Not a single one showed up at any other hoster during that period. A week later between May 18th and May 26th, 22 relays that contribute to relay spikes were deployed at Linode and during

[6] https://www.torproject.org/docs/tor-manual.html.en

[7] https://www.hetzner.com/

[8] https://www.linode.com/

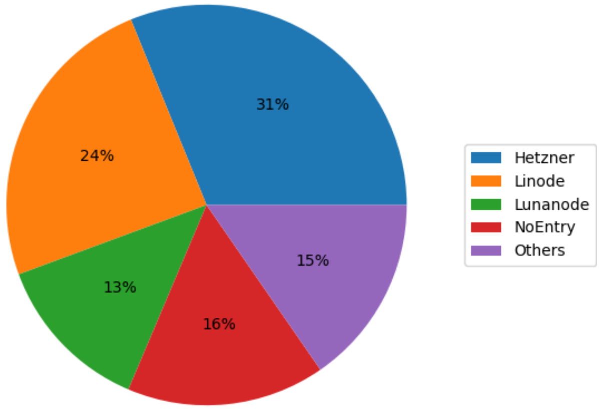

Figure 11: Provider assignment based on hostname for 462 relays that spiked at least 4 of 13 times

that period not a single one was deployed at Hetzner. While this is no conclusive proof, the probability of such a pattern emerging from random users deploying Tor relays seems negligible, even if there were a shared configuration source that is responsible for the monthly spikes.

The final observation we can contribute is that if we include relays that have been contributing to relay spikes at least 4 times (see Figure 11) a third cloud hosting provider, Lunanode[9], shows up with all relays having contributed to between 4 and 6 spikes. Combined with the fact that none of those relays are still running, this indicates that there was a third cloud hosting provider that was used in the past to operate this kind of relay. Unfortunately, we were unable to find any hints on what those relays are being used for and can therefore not tell if they are malicious, but the behavior definitely seems suspicious and will hopefully be further investigated by the Tor project.

4 CONCLUSION

Our analysis has found several inconsistencies within the Tor consensus that have the potential to negatively impact Tor users by limiting the amount of relays available to them without good reason. The most important aspect to improve upon would be to increase transparency on what specification directory authorities are currently employing. Just in the last year we have encountered multiple occasions where directory authorities clearly changed their voting behavior without publicly disclosing it or at least giving some indication of a change in their Tor version string. The Tor version strings themselves turn out to be insufficient because multiple directory authorities use self-compiled developer versions of Tor, where the version string tells us almost nothing about the actually running code. A potential improvement would be to include the commit hash and branch of the code in manually compiled Tor versions. This would still allow the Tor project to deploy hotfixes directly to directory authorities but keep transparency on when the running Tor version has changed. If the development branches are publicly visible, external analysts would even be able to find out if a deviation from the official directory specification is intended or not, which would greatly improve the transparency of the voting process.

Furthermore, we encourage a reevaluation of absolute flag criteria like the bandwidth required to obtain the Fast flag. The modern web is constantly developing and what was considered an acceptable bandwidth ten years ago, is no longer fitting today. In order for such flags to retain their usefulness, they should either drop requirements specified in absolute values or have a process in place to ensure they are updated regularly.

We also suggest searching for better ways to test potential directory specification changes. The current strategy of having a single directory authority voting differently from the others and occasionally handing those changes out to other authorities has already disrupted the Tor consensus more than once. One could either introduce a new directory authority that only creates internal vote previews without actually publishing votes or just have existing directory authorities log the data upon which they base their decisions. This would also help to avoid issues where a hotfix tested on one directory authority is accidentally bundled with voting behavior changes that were only intended for testing purposes.

Additionally, we believe that additional measures should be implemented to automatically detect suspicious spikes or patterns in the number of available relays. Malicious actors starting up a large number of Tor relays to launch attacks have been detected [10] in the past and situations where hundreds of relays join the network over a short period of time should be automatically detected by the Tor project and not go unnoticed for a year, especially if they stand out like this. Even if there is no sign that these relays are acting maliciously, the evidence hinting at those relays being run by the same entity should have been noticed by the Tor project earlier.

Finally, we would like to emphasize that our analysis has not found a single instance where we believe that malicious actors successfully modified the Tor consensus to attack users. Considering that Tor is a target for several nation state actors [1], we have to assume that attacks on the Tor consensus would have been launched if they were easy to execute. Despite the issues and improvement suggestions mentioned in this paper, it appears that Tor's method of forming a secure consensus in a globally distributed network does indeed withstand the test of time.

ACKNOWLEDGMENTS

This work has been carried out within the scope of Digidow, the Christian Doppler Laboratory for Private Digital Authentication in the Physical World, funded by the Christian Doppler Forschungsgesellschaft, 3 Banken IT GmbH, Kepler Universitätsklinikum GmbH, NXP Semiconductors Austria GmbH, and Österreichische Staatsdruckerei GmbH.

[9] https://www.lunanode.com/

REFERENCES

[1] James Ball, Bruce Schneider, and Glenn Greenwald. 2013. NSA and GCHQ target Tor network that protects anonymity of web users. https://www.theguardian.com/world/2013/oct/04/nsa-gchq-attack-tor-network-encryption

[2] Bram Cohen. 2008. The BitTorrent Protocol Specification. https://www.bittorrent.org/beps/bep_0003.html

[3] Roger Dingledine. 2021. Exit relay operators please help test #2667 branch. https://lists.torproject.org/pipermail/tor-relays/2021-January/019258.html

[4] Toralf Förster. 2021. Questions about consensus votes. https://lists.torproject.org/pipermail/tor-relays/2021-April/019604.html

[5] Sebastian Hahn. 2021. Questions about consensus votes. https://lists.torproject.org/pipermail/tor-relays/2021-April/019603.html

[6] Tobias Höller, Michael Roland, and René Mayrhofer. 2021. On the state of V3 onion services. In *11th Workshop on Free and Open Communications on the Internet (FOCI '21)* (Virtual Event, USA). ACM, 7 pages. https://doi.org/10.1145/3473604.3474565

[7] juga. 2019. How Bandwidth Scanners Monitor The Tor Network. https://blog.torproject.org/how-bandwidth-scanners-monitor-tor-network

[8] Karsten Loesing, Steven J. Murdoch, and Roger Dingledine. 2010. A Case Study on Measuring Statistical Data in the Tor Anonymity Network. In *Proceedings of the Workshop on Ethics in Computer Security Research (WECSR 2010)* (Tenerife, Canary Islands, Spain) *(LNCS)*. Springer.

[9] Tor metrics team. 2021. Tor metrics. https://metrics.torproject.org

[10] nusenu. 2021. say hi (and goodbye) to >1000 new exit relays at OVH. https://lists.torproject.org/pipermail/tor-relays/2021-May/019644.html

[11] The Tor Project. 2019. Tor directory protocol, version 3. https://gitweb.torproject.org/torspec.git/tree/dir-spec.txt

[12] Nakamoto Satoshi. 2009. Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.org/bitcoin.pdf

[13] sednet. 2013. TOR wants your spare bandwidth. https://www.linode.com/community/questions/8475/tor-wants-your-spare-bandwidth

[14] youiopmop. 2019. Setup a Tor relay on FreeBSD 12. https://community.hetzner.com/tutorials/setup-a-tor-relay-on-freebsd-12