Anonymity & Monitoring: How to Monitor the Infrastructure of an Anonymity System

Martin Mulazzani[1], Markus Huber[2], and Edgar Weippl[3]

[1] Security Research, Vienna, Austria `mmulazzani@securityresearch.at`
[2] Security Research, Vienna, Austria `mhuber@securityresearch.at`
[3] Vienna University of Technology, Austria `weippl@ifs.tuwien.ac.at`

Abstract. The Tor network is a widely deployed anonymity system on the Internet used by thousands of users every day. A basic monitoring system has been designed and implemented to allow long term statistics, provide feedback to the interested user and to detect certain attacks on the network. The implementation has been added to TorStatus, a project to display the current state of the Tor network. During a period of six months this monitoring system collected data, where information and patterns have been extracted and analyzed. Interestingly, the Tor network is very stable with more than half of all the servers located in Germany and the US. The data also shows a sinusoidal pattern every 24 hours in the total number of servers.

1 Introduction

This work is about monitoring the infrastructure of the Tor network. Tor, "The Onion Router", is an anonymity network that is widely used in the world to hide the user's IP address [12]. The required infrastructure is run by volunteers. Anyone can host a Tor server to improve the overall performance of the Tor network. Because servers can join and leave the current set of servers anytime, the network is highly dynamic and the overall performance depends on the time of usage. However, the number of servers is surprisingly constant over time.

To provide some form of monitoring of this widely distributed network is difficult as it is an anonymity network. Especially since the user's anonymity must not be compromised. We have implemented a form of monitoring to the infrastructure with a special focus on performance statistics, without introducing new or improving known attacks on the user's anonymity.

This work is organized as follows: The second section discusses anonymity on the Internet and the Tor network. Section 3 presents the approach chosen for the monitoring and the selected values of interest. Section 4 explains briefly the implementation, while Section 5 interprets the data from the monitoring over a period of six months. It shows interesting regular patterns and properties of the Tor network as well as unusual patterns. The last section is dedicated to the conclusion.

2 Anonymity on the Internet

The word anonymity comes from the Greek language, meaning "without a name" or "namelessness". When talking about anonymity, the usual setting refers to the transportation of a message from a sender to one or more recipients [29]. These messages can be of any kind (like electronic messages, votes in general elections, emails, etc.) and can be transported using any media (a piece of paper, radio waves, the Internet, natural language, etc.).

Anonymous electronic communication was first introduced in 1981 by David Chaum [5]. Instead of sending the message directly from the sender to the recipient it passes several additional relay servers on its way (so called mixes). These mixes collect the messages and forward them in batches. With the use of public key cryptography [10] it is possible that only the intended receiver can read the message and the mixes can not. Since the first introduction of mixes the development has evolved into two main directions: low latency and high latency anonymity systems [7].

High latency anonymity systems started in the early nineties and provided anonymity by mixing the messages over time. It is not important when exactly the message reaches the destination, but it has to reach it eventually with all possible identifying information removed. One of the first heavily used systems which started in 1992 was the pseudonymous remailer by Johan Helsingius (anon.penet.fi) [28]. By sending an email to this server including the intended target, it was possible to send *pseudonymous* emails and postings to the Usenet. However, in 1995 Helsingius was forced by the authorities to reveal a users real email address because of a complaint filed by the Church of Scientology. Thus the provided anonymity was weak, no encryption was used and the server was a single point of failure. If the server was down, the service was unusable. A deployed example for an anonymous remailer at the time of writing is Mixminion [8]. It was first published in December 2002 and is the successor of Mixmaster [21] and the Cypherpunk remailer [28]. It uses a set of trusted and synchronized directory servers which know the current network and server states. On the current Mixminion network there are between 80,000 and 100,000 messages transmitted daily [20]. The differences between the systems are that every succeeding remailer uses a stronger threat model and becames more resistant against attacks like traffic analysis [6]. One of the most recent developments in this field is the publication of the Sphinx cryptographic message format [9].

Low latency anonymity systems on the other hand are not able to use time to achieve strong anonymity. They are built to be usable for interactive communication like SSH or web browsing, which means that long delays like in anonymous remailers are unacceptable. Bidirectional communication is used where a client can request something from another entity like a server and gets the response with only minor additional delay. Various systems exist with different threat models and different compromises between flexibility and anonymity. One of the first systems designed was “ISDN-mixes” [30]. Today, Tor is probably the most heavily used with an estimated 100,000 plus users every day. An alternative to Tor is “JonDo”, formerly known as the Java Anon Proxy [4].

2.1 The Tor Network

Tor [12] is the second generation of onion routing and is based on the original onion router [15] from 1999. It is a low latency anonymity system and used all over the world. The public network was deployed in October 2003 [13] and has grown from only a few servers to more than 2000 distinct servers with around 1200 servers running any time at the time of writing. It is an open source project and offers a lot of features compared to earlier implementations of onion routing for users seeking anonymous communication.

The goal of the Tor network is to provide communication anonymity and to prevent traffic analysis [6], thus making it hard for an attacker to link communication partners or link multiple conversations to or from a single user. It was developed for a high degree of deployability and usability, resulting in an increased anonymity set [11]. Tor also makes it easy for anyone to extend the underlying infrastructure by running a Tor node at very low cost. Tor nodes run on a variety of platforms and the system is very flexible regarding what applications can be used, and incorporates many good designs from previous anonymity systems.

The threat model of Tor is weaker than other anonymity systems. The most commonly assumed threat regarding anonymity systems is a global passive adversary who can monitor all traffic going into and out of an anonymity systems as well as the traffic inside of the system. Additionally, it is often assumed that the adversary may has the ability to inject, delete or modify messages in transit and a subset of the nodes can be under this persons control; it is however unrealistic that he controls all of the nodes. Anonymity systems that protect against such attackers can assume to be secure for most of the possible users [12]. By design Tor does not protect against such a strong attacker. The assumed adversary in Tor is able to run or control a subset of network nodes, is able to observe some fraction of the network traffic and is able to inject, delete or modify messages [12].

Tor protects against traffic analysis attacks which would allow an attacker to find out which nodes of importance to attack only by watching traffic patterns. In favor of a simple and deployable design, some commonly used techniques in anonymity like message reordering were intentionally not used when Tor was designed. The network is not solely peer-to-peer based and is not using steganography; thus it is not hiding the fact that somebody is using Tor. Another design choice was to keep Tor completely separated from protocol normalization: many protocols like HTTP are very complex and variable, which would require complex protocol normalization to make all clients look the same. For HTTP, Tor relies on TorButton, a firefox extension to prevent IP and cookie leakage, and

Privoxy, a filtering web proxy that enhances privacy. TorButton on the other hand filters many types of active web content which could possibly leak identity. For other protocols this has to be considered separately before relying on the anonymity provided by Tor.

The Tor network consists of the following basic entities - the details of the basic architecture and the entities can be found in the original paper [12] as well as on the Tor website [1]:

– Directory Server: The core of the Tor network, a small set of trusted authorities (7 at the time of writing). They know the current set of valid Tor servers, distribute this knowledge to the clients and sign it with their private keys. The directory servers vote on the current set of Tor servers with simple majority, so that control over one directory server has no value. For an adversary control over more then half of these servers would be required to successfully attack Tor at the directory server level.
– Server: The Tor servers are used to actively hide the user's IP address. They are operated by volunteers, anyone can download the server software and run a Tor server. As soon as the server program is started it publishes its keys and descriptor to the directory servers, and will then become part of the directory. Once the server is listed in the directory it can be used by clients. It will continue to publish signed statements to the directory servers constantly. To be as flexible as possible, every server operator can decide to limit the bandwidth or transferred data Tor might consume, and whether the server will be an exit server and allow connections to hosts outside the Tor network. There are many other configurable options and many properties a server can have, for example if it is flagged as a guard node. Guard Nodes [34] were not part of the original design but were added later because of a certain attacks [25] against hidden services and predecessor attacks in general [37].
– Clients: The client software runs on the user's computer. Right now there are two software clients available, the original client written in C and available for the majority of popular operating systems (Windows, Mac OS X and various Linux/Unix flavors) and OnionCoffee, which was written in Java for the PRIME project [31] as a proof of concept and is no longer maintained. Additionally the client is available in different packages, like preconfigured USB sticks and VMware images.

Communication in Tor works on the transport layer, anonymizing TCP streams. Many TCP streams can share a circuit, and by default a circuit consists of three Tor servers: the *entrance node*, the *middleman node* and the *exit node*. The entrance node is aware of the real IP address of the users, but not the communication content whereas the exit node is able to see the content the user was originally transmitting and the final destination but not the originating IP address. The circuit is constructed incrementally by the client software. Everything is transported in fixed size cells, which get encrypted once for every relay in the circuit by the client. Packet anonymization instead of stream anonymization was proposed [17] and has the drawback of causing a lot of resend requests as the packets take different paths with arbitrary delays to the destination.

As Tor became popular in recent years, many attacks were designed to defeat or degrade anonymity [3, 16, 22] and their according countermeasures [25]. Among the most notable attacks is the Sybil attack [14], which is applicable on almost any system that relies on distributed trust. It is based on the idea that a small number of entities can impersonate multiple identities. Traffic analysis [2] on the other hand uses metadata from network communications instead of content to attack anonymity systems, as content is often encrypted with strong ciphers. This metadata includes volume and timing information as well as source and destination of messages. Tor does not protect against a global passive adversary that can monitor all network connections. Traffic analysis makes it possible to link communication partners at random, just by controlling and monitoring some part of the network. This has been shown in [23] by estimating the traffic load on specific Tor nodes. Other publications wanted to improve the overall Tor performance and the security inside Tor [24, 26, 27, 32, 33].

TorStatus, also known as Tor Network Status, is one of many open source projects around the Tor Network [35]. Its goal is to present the current status of the Tor network to the interested user. In essence it consists of a Perl script which connects to a local Tor server and writes the necessary information into a database, the frontend is a PHP application which takes care of the appropriate presentation. It was originally developed by Joseph Kowalski in 2006. The current lead developer is Kasimir Gabert.

3 Monitoring Tor

Monitoring the Tor network has been done before, but with a different focus. Researchers tried to find out more information about Tor users and to understand what they use Tor for [19] in the beginning of 2008. This is somehow contradicting with the purpose of an anonymity service. Usage pattern on the other hand are important for future development of Tor and the perception of Tor in the public. The result was that most connections are used for HTTP traffic, while most of the traffic is caused by BitTorrent, a common file sharing protocol. For Tor hidden services, statistics have been collected as well [18], but without long term public presentation.

3.1 Values of Interest

Before monitoring a system it is important to identify the values of interest, i.e. not all values are bearing useful information. As with the Tor network it is important to find values that have a direct influence on the user's anonymity. Care has to be taken not to introduce possibilities for new attacks on the network, or to decreasing the degree of anonymity and making existing attacks easier. The Tor infrastructure is a dynamic number of Tor servers operated by volunteers, so the most basic values of interest consists of the total number of Tor servers.

Monitoring the infrastructure of the Tor network has been done before but was suspended again. There is a website that collected the number of Tor nodes and the total traffic of the Tor network over time, [36], but stopped publishing those values as of August 2007. Instead, it refers to the current TorStatus websites, which only shows the current state of the network and does not monitor anything over time (yet). One of the design papers by the creators of Tor [13] published some statistics about the number of running routers and the total relayed traffic in the early stages of the Tor network, from August 2004 till January 2005. Apart from that no official statistics over time are publicly available.

Inspired by those early data collections the following important values worth monitoring have been identified:

- Total number of running servers
- Total number of running exit servers
- Total number of running fast servers
- Total number of running guard servers
- Country distribution of running servers
- Country distribution of running exit servers

These values have been chosen with two goals in mind: to measure the effects of the infrastructure on the possible degree of anonymity and to measure the effects on the quality of service.

For the overall anonymity and security, the number of Tor servers at the time of usage is crucial. A handful of servers would be easy to attack, either directly with the goal of defeating anonymity or with a denial of service attack. Thus the greater the number of Tor servers, the better. In the network consensus all servers are flagged according to their capabilities and qualities. The important flags for the degree of anonymity are "running" and "exit server". Those are the servers a client uses for his circuits, as apparently a server that is not reachable is of no value in a Tor circuit. Thus the number of running servers and running exit servers is the smallest practical metric with influence on the path selection and anonymity. The client's actual path selection algorithm is more complex and depends on further server attributes like advertised bandwidth and server IP address, which are not part of the monitoring. For the quality of the Tor connections, two other flags become important, namely the "fast" and "guard" nodes: the more fast Tor nodes there are, the better the overall performance for the client. The more guard nodes in the network, the better the performance (as the proportion guard nodes to clients decreases) and the number of clients a given guard handles decreases.

The distribution of servers over different countries and therefore different jurisdictions is the second value that influences the degree of anonymity. To keep the demands on the performance of the monitoring system low and to prevent the user from an uncontrollable information flood, only selected countries are getting monitored. The countries with the highest number and the vast majority of Tor servers are those of interest. Over a period of two months the country distribution of Tor servers has been monitored, resulting in the implementation of the following set of countries to monitor: United States of America, Germany, China, France, Sweden, Russia, the Netherlands, Canada, Great Britain, Italy and Austria. Additionally the sum of all other countries and Tor nodes that are not mappable to any country get accumulated and stored as well. Based on this information, the top 11 countries have been selected for monitoring. The mapping of IP addresses to countries was done by using TorStatus, more precisely by using the GeoIP library together with TorStatus.

4 Implementation

The gathering of data was implemented by expanding the TorStatus update script, *tns_update.pl*, together with a graphical user presentation in the frontend, *network_history.php*. Whenever the script is updating the TorStatus database, it is also updating the Tor History monitoring. The values are stored in a RRD, a so called round robin database, which is part of the software "RRDtool" and was already used for generating and updating the other graphs of the TorStatus website. The changes can already be found in the SVN trunk of TorStatus, and will be included in the next release.

When working on an anonymity network it has to be assured that by adding features the additional features should never possibly harm the network or anonymity. This means that the new features should not introduce new attacks or make existing attacks easier. In our case, all the used information is part of the network consensus, which means that all the information are already known to every Tor server as well as every Tor client, they are kind of public knowledge. The only difference is that some coarse part of it is now stored over time. From all the recent published attacks on the Tor network none becomes easier. And even future attacks should not benefit from it, as e.g. the number of servers per country or the total number of exit servers holds no valuable information when trying to attack the user's anonymity. The main objective is to provide the interested user an overview over the current status of the underlying Tor network and to make changes in the Tor network visible.

4.1 Further improvements

The collected data set might be expanded to collect information of other countries in detail, and collect additional data on the overall network. One useful example would be the total transmitted traffic (as in [13]), but as this information is provided by the servers it is easy to manipulate and to bias the statistics. Another feature worth implementing in the future would be the port distribution of the exit servers in general and the distribution of specific important ports like

HTTP by country. As soon as IPv6 is supported by Tor the monitoring software should be extended to support IPv6 as well.

The focus of this work was on basic network monitoring, so these improvements are left open for the future. However, any extensions should be done very carefully and only after evaluating the impact on anonymity.

5 Data Interpretation

The data collection started in the middle of October 2008 and is still going on. The time interval of the collected data presented here refers to around six months. Note that the time zone in the graphics is GMT and that gaps in the graphics were caused by failures in the data collection process.

5.1 Overall Network Size

Since deployment, the overall network size and the number of running Tor servers has been steadily increasing. Since public deployment in October 2003 with only a handful of servers, the infrastructure has grown to a total of 50 nodes by November 2004 [23]. This doubled to around one hundred nodes in January 2005 [13]. In August 2007 there were around 1000 running servers as of [36]. Up to today, everyday between 1300 and 1500 servers form the basic infrastructure of the Tor network, whereof 600 to 700 of them are exit servers.

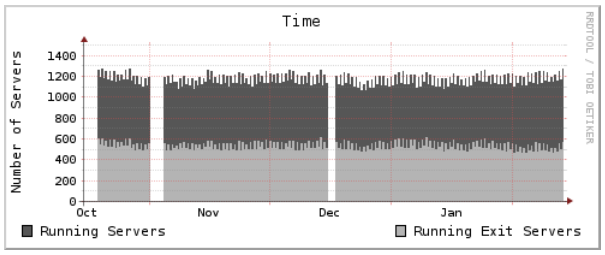

Fig. 1. Running Tor Servers in the first 3 months

Figure 1 shows the overall number of servers and the number of exit servers during the first three months of the data collection period. On average, 1163 servers were online during that period, 532 of them exit servers. The highest number of servers were 1289, the lowest 1040 (626/454 exit servers). It is clearly visible that the number of servers stays roughly the same and the network is quite constant. This does not mean that these are the same servers all the time: more then half of the servers that report their uptime have an uptime less then one week. Less then five percent have an uptime which is longer then ten weeks.

About eighty percent of the servers have a reported uptime in TorStatus. During the last 3 months the overall network size increased, which is shown in figure 2.

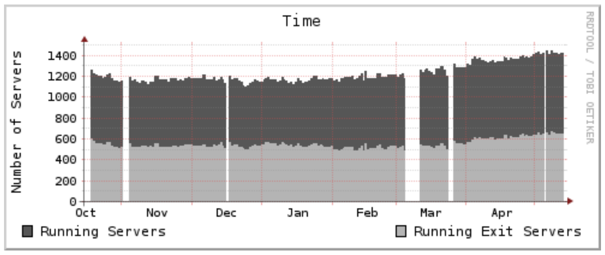

Fig. 2. Running Tor Servers during 6 months

On country level the most active countries are by far Germany and the United States. Both countries together host in total more than half of all the servers, which is shown in figure 3. The reasons for that are not easy to identify. Among

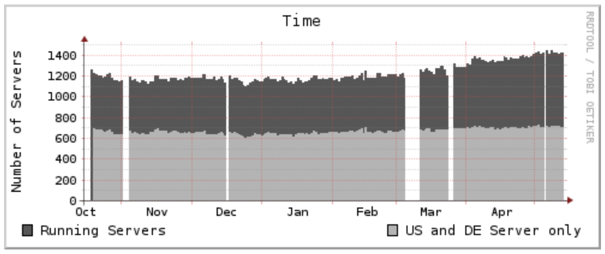

Fig. 3. Running Servers in DE and US, in total

other reasons, data privacy seems to have a high significance in Germany, probably based on the country's history and because of many popular organizations that want to enforce civil rights and electronic privacy like the Chaos Computer Club. In the US there is a different approach on data privacy compared with Germany or the European Union. Both countries are among the top five countries regarding the Internet density in the world according to internetworldstats.com. If the Internet density is high, Internet access and bandwidth for the users becomes cheap. This of course increases the likelihood that a user is willing to donate bandwidth to the Tor project by running a Tor node.

5.2 Total Number Patterns

The most obvious pattern is that the number of overall servers is dependent on the current time of day. This is shown in figure 4. It looks like a sine function (or

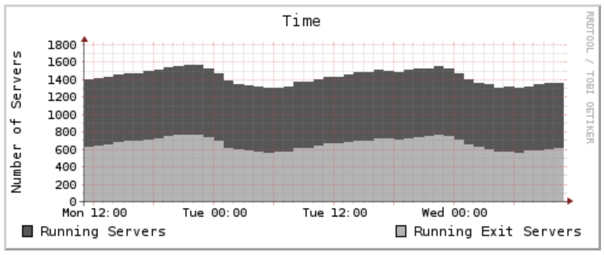

Fig. 4. Running Tor servers within 48 hours

cosine) as the number of servers reaches a maximum and a minimum within a 24 period each day. The time in the figure is GMT, which means that most of the servers are online in the "GMT evening" which is insofar interesting as this data represent the worldwide Tor network. This is mainly influenced (among other things) by the daily variation of servers in Germany, where most of the servers are online in the evening.

This daily pattern is also found in the data set of the "other" servers. These are the servers were the country is unknown, summed up with the servers that are in a different country then the countries monitored separately. The sum of the other servers and how it changes over time is shown in figure 5. By looking

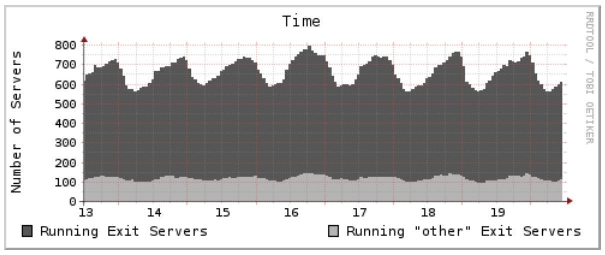

Fig. 5. Daily pattern in "other servers"

at it, it is visible that this category of servers as well has a daily pattern. The amplitude is much lower than compared with the number of exit servers, but it is still visible. This means that firstly Germany seems to be the reason that the number of servers changes over the day, but also the servers that are not in the top countries or where the country is not identifiable by GeoIP within TorStatus. Secondly, it seems that the majority of servers have to be geographically close if the server administrators operate their servers similar, which is needed that an overall pattern becomes visible.

Finally, regarding the overall network pattern, the proportion of exit servers is of importance. The Tor network is designed that 33 % of the servers need to be exit servers. Figure 6 shows that between 45 % and 50 % of the servers are exit

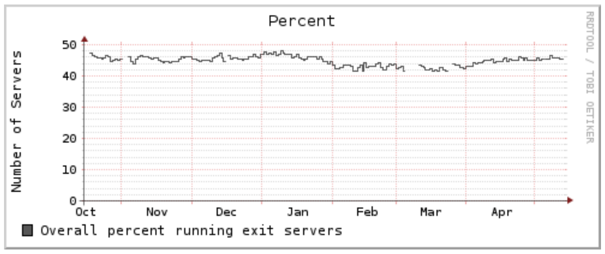

Fig. 6. Overall percent exit servers

servers, which is very good. Just because a server is flagged as an exit server does not mean that the client software has to choose this server as the exit server. Interestingly, the ratio of normal to exit servers stays quite constant compared to the rest of the Tor network. Sudden changes in the percentage of exit servers would be a good indicator in case of any attacks. For example if an attacker tries to redirect as many users as possible to his exit servers by attacking the others with a denial of service attack. However, a sophisticated attacker might combine the denial of service attack with a Sybil attack and start an exit server for every attacked exit server. This would keep the ratio unchanged. The percentage of exit servers does not take into account the various exit server policies as they allow services based on the port number, but gives a good overview of the overall network.

In our monitoring, around 80 % of the servers are monitored, which means that at least 80 % of all Tor servers are hosted in only 11 countries. In the future it would make sense to take the various exit policies into account as well.

5.3 Country Pattern

When watching the data on country level, other interesting patterns can be identified. The patterns discussed in this section refer to the top countries that are monitored separately. The biggest part of the Tor network is made by servers from Germany and the United States. By comparing those two countries, subtle differences are detectable. This is shown in figure 7. While the number of servers in the US stays quite constant, the number of servers in Germany is changing with a 24 hour pattern. Around 50 servers or 15 % of the German servers are turned off and on again during the night. This might be because of Internet access pricing models from Internet service providers or because the servers are operated at workplaces. The exact reasons are hard to tell. The total number of

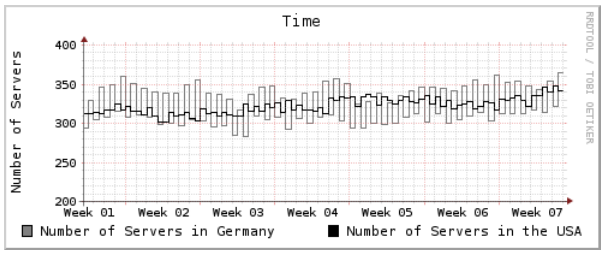

Fig. 7. Differences Germany and the USA

servers in Germany per day is between 300 and 400.

It is also interesting how many servers in a country are exit servers. This is shown for a few countries in figure 8. Only the top 5 countries have been

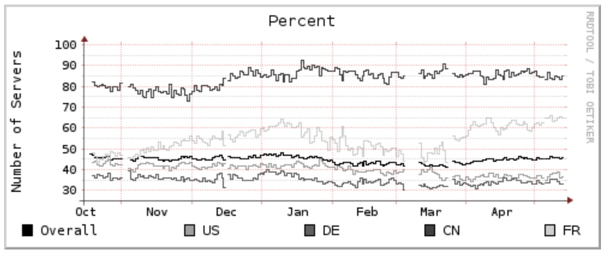

Fig. 8. Percent of Exit Servers

compared for simplicity of the graph. It is very interesting to see which countries have more than average in exit servers. The average of exit servers was already shown in figure 6. Germany and the United States have a less then average percentage of exit servers, while the countries hosting not so many Tor servers like China and France have a higher percentage. This is most likely because of the much lower number of servers. Overall, China has constantly between 80 and 90 % exit servers.

5.4 Strange Pattern found

During the collection of data, some strange patterns have been found. One of them seems to be inside of Tor itself: When looking at the number of guard servers, the values change quite a lot. It seems as if the Tor consensus is adding and removing a large proportion of guard servers. This is shown in figure 9. Although the number of guard servers seems to change a lot, there is still a daily visible pattern. The sine pattern again can be attributed to Germany, the country with the highest number of guard servers, and some other countries.

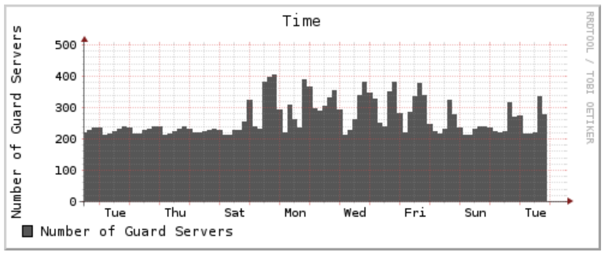

Fig. 9. Problem with number of Guard Servers

The large jumps with more than 100 guard nodes and more joining and leaving the set of guard nodes is not explicable by daily pattern. This is likely due to some topics in the Tor implementation, or the network consensus protocol. This issue has been reported to the Tor developer team and not yet solved.

6 Summary and Conclusion

Monitoring the Tor network greatly increases the overall benefits of the Tor community. First, the most obvious benefit, is that the development and growth of the Tor network is now visible to the Tor developers, the server operators, the users and everyone interested. The information is presented conveniently and self-explanatory, no deep knowledge or understanding of the Tor network is needed. This feedback can be used by the advanced user to decide about the current degree of anonymity and whether the network is providing sufficient anonymity or not. The collected statistics about the history of the Tor network might one day allow decisions as to which country is to be used as an entry to Tor or other improvements to the path selection. An example: all of the sudden the number of servers in Italy drops significantly. Is it then still ok to use the remaining servers in Italy? It probably is ok for a user relatively close to Italy who is seeking high data throughput, if the Italian servers are used as a middleman or entry node. For a user seeking high anonymity it might be better to avoid Italy in all circuits at all. Recently the complete network consensus information since the beginning of the Tor network has been made available, which will further improve the findings of this paper in the future. By comparing the present with the past it becomes possible to observe trends and this might be a triggering point for an analysis or discussion.
This brings us to a very important benefit, the possibility of attack detection. Right now, only the server operator is able to detect attacks on servers under his control, like denial of service attacks. Sometimes there follows a discussion on the Tor mailing list (or-talk@freehaven.net), mainly to check if other operators have or had similar problems. If an attacker would launch a more significant attack like a distributed denial of service attack on half of the Tor servers, the real impact on the Tor infrastructure would be hard to detect fast. The same holds if

an attacker launches a Sybil attack by adding numerous potentially manipulated servers. With the monitoring of the Tor network this becomes easier to detect, as the change in the number of servers is reflected immediately respectively as soon as the change becomes part of the consensus.

In the far future, the collected information could be used to compare the current state of the network with the expected state. By calculating a conservative set of rules about the "normal" behavior of the Tor network, changes that could probably have an influence on the security can be detected. Incorporating these rules into the path selection algorithm would allow finely tuned privacy constraints on the selected nodes. An example would be an adaptive path length: in the normal case, three nodes could get selected as it is currently implemented. If then all of the sudden the number of servers drops or significantly increases the path length could be automatically set to six Tor nodes to increase security. It is not clear if an increase in path length would increase the security, but it is sufficient as an example where the collected data of the Tor network could get used in the future. Another example would be the integration of the data into TorFlow, a project to detect malicious nodes. Instead of testing all the servers all the time, which could eventually be costly in terms of time and resources, only selected Tor nodes or all the nodes in a specific country could be tested. Based on the collected data a set of Tor nodes responsible for the most suspicious changes in the infrastructure could be identified for testing.

Acknowledgements

We would like to thank the anonymous reviewers. This work has been supported by the Austrian Research Promotion Agency under grant 820854.

References

1. Tor: anonymity online. `https://www.torproject.org/`.
2. Adam Back, Ulf Möller, and Anton Stiglic. Traffic Analysis Attacks and Trade-Offs in Anonymity Providing Systems. In *Proceedings of Information Hiding Workshop (IH 2001)*, pages 245–257, April 2001.
3. Kevin Bauer, Damon McCoy, Dirk Grunwald, Tadayoshi Kohno, and Douglas Sicker. Low-Resource Routing Attacks Against Tor. In *Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2007)*, October 2007.
4. Oliver Berthold and Hannes Federrath. Project "anonymity and unobservability in the internet". *Workshop on Freedom and Privacy by Design / CFP2000*, 2000.
5. David Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communications of the ACM*, 4(2), February 1981.
6. George Danezis. Introducing Traffic Analysis: Attacks, Defences and Public Policy Issues. Technical report, University of Cambridge Computer Lab, 2005.
7. George Danezis and Claudia Diaz. A Survey of Anonymous Communication Channels. Technical report, Microsoft Research, January 2008.

8. George Danezis, Roger Dingledine, and Nick Mathewson. Mixminion: Design of a Type III Anonymous Remailer Protocol. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, pages 2–15, May 2003.
9. George Danezis and Ian Goldberg. Sphinx: A compact and provably secure mix format. *Security and Privacy, IEEE Symposium on*, 2009.
10. Whitfield Diffie and Martin E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.
11. Roger Dingledine and Nick Mathewson. Anonymity Loves Company: Usability and the Network Effect. In *Proceedings of the Fifth Workshop on the Economics of Information Security (WEIS 2006)*, June 2006.
12. Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The Second-Generation Onion Router. In *Proceedings of the 13th USENIX Security Symposium*, August 2004.
13. Roger Dingledine, Nick Mathewson, and Paul Syverson. Challenges in deploying low-latency anonymity (DRAFT), February 2005. `https://www.torproject.org/svn/trunk/doc/design-paper/challenges.pdf`.
14. John R. Douceur. The Sybil Attack. In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, 2002.
15. David Goldschlag, Michael Reed, and Paul Syverson. Onion Routing for Anonymous and Private Internet Connections. *Communications of the ACM*, 42:39–41, 1999.
16. Andrew Hintz. Fingerprinting Websites Using Traffic Analysis. In *Proceedings of Privacy Enhancing Technologies workshop (PET 2002)*, April 2002.
17. Olaf Landsiedel, Alexis Pimenidis, Klaus Wehrle, Heiko Niedermayer, and Georg Carle. Dynamic Multipath Onion Routing in Anonymous Peer-To-Peer Overlay Networks. In *Proceedings of the GLOBECOM 2007*, 2007.
18. Karsten Loesing, Werner Sandmann, Christian Wilms, and Guido Wirtz. Performance Measurements and Statistics of Tor Hidden Services. In *Proceedings of the 2008 International Symposium on Applications and the Internet (SAINT)*, July 2008.
19. Damon McCoy, Kevin Bauer, Dirk Grunwald, Tadayoshi Kohno, and Douglas C. Sicker. Shining Light in Dark Places: Understanding the Tor Network. In *Privacy Enhancing Technologies*, pages 63–76, 2008.
20. Mixminion network load. `http://www.noreply.org/load/`.
21. Ulf Möller, Lance Cottrell, Peter Palfrader, and Len Sassaman. Mixmaster Protocol — Version 2. Draft, July 2003. `http://www.abditum.com/mixmaster-spec.txt`.
22. Steven J. Murdoch. Hot or Not: Revealing Hidden Services by their Clock Skew. In *Proceedings of CCS 2006*, October 2006.
23. Steven J. Murdoch and George Danezis. Low-Cost Traffic Analysis of Tor. In *Proceedings of the 2005 IEEE Symposium on Security and Privacy*. IEEE CS, May 2005.
24. Steven J. Murdoch and Robert N. M. Watson. Metrics for Security and Performance in Low-Latency Anonymity Networks. In *Proceedings of the Eighth International Symposium on Privacy Enhancing Technologies (PETS 2008)*, July 2008.
25. Lasse Overlier and Paul Syverson. Locating Hidden Servers. In *SP '06: Proceedings of the 2006 IEEE Symposium on Security and Privacy*, pages 100–114. IEEE Computer Society, 2006.
26. Lasse Øverlier and Paul Syverson. Improving Efficiency and Simplicity of Tor Circuit Establishment and Hidden Services. In *Proceedings of the Seventh Workshop on Privacy Enhancing Technologies (PET 2007)*, June 2007.

27. Andriy Panchenko, Lexi Pimenidis, and Johannes Renner. Performance Analysis of Anonymous Communication Channels Provided by Tor. In *ARES '08: Proceedings of the 2008 Third International Conference on Availability, Reliability and Security*, pages 221–228. IEEE Computer Society, 2008.
28. Sameer Parekh. Prospects for Remailers. *First Monday*, 1(2), August 1996. `http://www.firstmonday.dk/issues/issue2/remailers/`.
29. Andreas Pfitzmann and Marit Hansen. Anonymity, Unobservability, and Pseudonymity: A Consolidated Proposal for Terminology. Draft, February 2008. `http://dud.inf.tu-dresden.de/Anon_Terminology.shtml`.
30. Andreas Pfitzmann, Birgit Pfitzmann, and Michael Waidner. ISDN-mixes: Untraceable communication with very small bandwidth overhead. In *Proceedings of the GI/ITG Conference on Communication in Distributed Systems*, pages 451–463, February 1991.
31. PRIME - Privacy and Identity Management for Europe. `https://www.prime-project.eu`.
32. Stephen Rollyson. Improving Tor Onion Routing Client Latency. Technical report, Georgia Tech College of Computing, 2006.
33. Robin Snader and Nikita Borisov. A Tune-up for Tor: Improving Security and Performance in the Tor Network. In *Proceedings of the Network and Distributed Security Symposium - NDSS '08*, February 2008.
34. TOR Path Specification. `https://svn.torproject.org/svn/tor/trunk/doc/spec/path-spec.txt`.
35. TOR Network Status Project. `http://project.torstatus.kgprog.com/`.
36. TOR Running Routers, outdated as of August 2007. `http://www.noreply.org/tor-running-routers`.
37. Matthew Wright, Micah Adler, Brian Levine, and Clay Shields. The predecessor attack: An analysis of a threat to anonymous communications systems. *ACM Trans. Inf. Syst. Secur.*, 7(4), 2004.