Tor HTTP usage and Information Leakage

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

[1] Vienna University of Technology, Austria mhuber@sba-research.org
[2] Secure Business Austria, Austria mmulazzani@sba-research.org
[3] Vienna University of Technology, Austria weippl@ifs.tuwien.ac.at

Abstract. This paper analyzes the web browsing behaviour of Tor users. By collecting HTTP requests we show which websites are of interest to Tor users and we determined an upper bound on how vulnerable Tor users are to sophisticated de-anonymization attacks: up to *78 %* of the Tor users *do not* use Tor as suggested by the Tor community, namely to browse the web with TorButton. They could thus fall victim to de-anonymization attacks by merely browsing the web. Around 1% of the requests could be used by an adversary for exploit piggybacking on vulnerable file formats. Another 7 % of all requests were generated by social networking sites which leak plenty of sensitive and identifying information. Due to the design of HTTP and Tor, we argue that HTTPS is currently the only effective countermeasure against de-anonymization and information leakage for HTTP over Tor.

Keywords: Tor, information leakage, privacy

1 Introduction

The Tor network [1] is a widely deployed anonymization network which hides the user's IP address on the Internet. It is expected to be used by hundreds of thousands of users every day and is the most heavily used open anonymization network today [2]. The main contribution of this paper is an in-depth analysis on how the Tor network is used to browse the web.

At the time of writing little is known about the traffic that leaves the Tor anonymization network. McCoy et al. [3] published a first investigation into how Tor is being used by its end-users. They found that the majority of connections leaving the Tor network are caused by the hypertext transfer protocol (HTTP). According to statistics from the Tor project [4], their anonymization network played a crucial role in the aftermaths of the latest Iranian elections whereas HTTP-based services such as Facebook had been blocked on a national-level. The ability of Tor to bypass Internet-censorship techniques got recently even the attention of mainstream media (e.g. Forbes [5]). A deeper analysis of HTTP traffic that is tunnelled trough the Tor network is however lacking. Hence we aim to provide a first investigation into how Tor is used to access the world wide web.

By running a Tor exit server and logging the HTTP requests, we collected in total 9 x 10^6 HTTP requests over a period of several weeks. The captured HTTP requests form the basis of our investigation into the web browsing behaviour of Tor users. The main contributions of this paper are:

- An in-depth analysis of HTTP traffic tunnelled through the Tor network (Section 4).
- HTTP-based attack scenarios and mitigation strategies (Section 5).
- Potential risks for complete HTTP user de-anonymization, which can not be prevented by Tor (Section 5.1).

We additionally show dangers to which users are exposed by using Tor incorrectly and which information users leak by browsing the web with Tor.

The rest of the paper is organized as follows: Section 2 discusses Tor and previous work about Tor usage analysis. Section 3 shows how we collected our data while we interpret the results in Section 4. Section 5 describes new attack scenarios based on HTTP as well as countermeasures. The last section is dedicated to the conclusion.

2 Tor Security and Threat Model

The Tor network hides the user's IP address by sending its packets through a number of Tor servers. Tor itself does not hide nor encrypt communication content leaving the Tor network: the user has to take care that it is used correctly. Sensitive information should only be sent over an encrypted protocol such as HTTP secure (HTTPS). A passive adversary running an exit server would need to break the end-to-end encrypted transmission in order to capture sensitive information. We will show later to what extent sensitive information is transmitted unencrypted over HTTP.

The basic infrastructure of Tor is run by volunteers, and anyone can set up a relay at relatively low cost. It provides reliable, bi-directional communication that can be used for low latency communication such as interactive conversations, shell access or web browsing. Tor can be used by any program that is able to use a SOCKS proxy and is freely available for almost any operating system as well as in the form of prepared live CDs and virtual machines. Tor uses three servers per path by default to hide its users real IP address, all servers chosen at the client-side. The user's communication content is protected from eavesdropping within the Tor network by using multiple layers of encryption. Before forwarding a message, every Tor server removes his layer of encryption.

At the time of writing, the Tor network consists of approximately 1600 running servers, distributed all over the world. The first server in a path is chosen from an ordered set of so called "entry nodes"; these servers are able to see the users real IP address. Without the ordered set, every entry node would eventually

see every user's real IP address. The last Tor server in the path is the so called "exit server" and is chosen based on the communication target's port number and self-proclaimed available bandwidth. The so called "exit policy" at every server specifies which port numbers are allowed for communication and whether the server is allowing connections only within Tor or leaving Tor to the regular Internet. The exit policy is defined by the server operator. Finally, a small set of trusted "directory servers" collect information about the current state of the network and vote on the current set of reachable servers. This information is publicly available to all clients. The security of the Tor network relies on the fact that instead of a single point or entity a user has to trust (for example by using an open proxy server or a dubious VPN service); the trust is distributed among the three Tor relays in a path.

Previous research about the usage of Tor has been conducted in the beginning of 2008 [3]: by running a high bandwidth Tor relay and inspecting the communication content it was found that the majority of connections leaving Tor was created by HTTP traffic, in total more than 90 %. However, a disproportional part of the transferred data was caused by BitTorrent, a common file sharing protocol. Yet a detailed analysis of the HTTP usage has not been conducted. Another analysis has been conducted by Dan Egerstad in 2007 [6] who published a list of 100 sensitive email accounts including passwords from embassies that apparently used Tor incorrectly. Other published attacks on Tor aimed at decreasing or defeating the users anonymity by means of traffic analysis [7,8,9,10] as well as attacks on unique aspects such as path selection [11] or the "hidden services" of the Tor network [12,13].

3 Tor HTTP Sniffing - Ethics and Methodology

In our experiment we ran a Tor exit node and collected all HTTP requests by running *urlsnarf* from the *dsniff* toolkit [14]. Urlsnarf sniffs HTTP traffic and is able to format it in CLF (Common Log Format), which is a format commonly used by webservers. Compared to other experiments [3] our server was advertising less bandwidth to represent an average node and not to bias the client's Tor path selection algorithm towards our node; only HTTP traffic was allowed in our exit policy. The collection period was from December 2009 till January 2010 with 9 x 10^6 HTTP requests in total resulteding in a logfile of 2.5 gigabytes. We took special care that the users identities were not endangered or revealed during the collection process: we did not store any packet dumps, user credentials, authentication cookies or any other possibly compromising data except the HTTP request. We did not try to become a "guard server" which clients use as their entry to the Tor network and which are able to see the users real IP address, and we did not monitor incoming connections. The data was stored in an encrypted filecontainer and moved to another computer after the collection process to protect against data disclosure in case of a server compromise.

A Python script using the "apachelog" libary [15] deconstructed the requests into three parts, according to our evaluation criteria:

- *Target domain*: the domain name of the request, without any deeper analysis of the specific requested file or session context.
- *User agent*: the string that the browser sends to identify itself. This gives a hint if the users are using TorButton, the recommended software by the Tor developers to prevent user identification by an adversary.
- *File type*: the extension of the requested file. Executables pose a direct danger to the user as an adversary might replace or modify them to defeat anonymity or even worse. Indirect danger comes from file formats were there exist known vulnerabilities in the corresponding software.

Subsequently the requests were sanitized and normalized for evaluation:

- File formats: various file extensions are treated the same on the client side, notably image file formats such as `jpg` and `jpeg` or websites within the browser, such as `html`, `htm`, `php` or `cgi`.
- Domain affiliation: some websites use different domains for content distribution, such as for example `fbcdn.net` and `facebook.com` belong to the same website.

4 Results & Data Interpretation

Our goal was to analyze what Tor is used for, which domains are most popular among Tor users and to discover potential threats to users.

4.1 Domains

In a first analysis based on the collected HTTP traffic we looked into depth into the different websites that were visited through our Tor exit server.

Table 1a shows the top 10 visited domains, based on the percentage a certain domain accounts to the total requests. Facebook.com and google.com were amongst the most visited sites. In fact, the majority of the requests belonged to one of the following website categories:

- *Social networking* sites (e.g. facebook.com, blackplanet.com)
- *Search engines* (e.g. google.com, yellowpages.ca)
- *File sharing* (e.g. btmon.com, torrentbox.com)

Social networking sites (SNSs), which today account to the most popular web sites, account in total to 7.33 per cent of all analysed HTTP traffic. These web-services are interesting for two reasons: SNSs leak plenty of personal information and secondly these services do not support HTTPS at the time of writing, except for user authentication. Table 1b shows the Top SNSs as well as how much of the SNSs traffic accounts to which service.

Domain (total %)	
URL	Per cent (%)
'facebook.com'	4.33
'www.google.com'	2.79
'blackplanet.com'	1.61
'yandex.ru'	1.57
'btmon.com'	1.47
'photobucket.com'	0.98
'craigslist.org'	0.90
'torrentbox.com'	0.88
'ameba.jp'	0.87
'maps.google.com'	0.70

(a) Overall services

Social Networking Site (relative %)	
Name	Per cent (%)
'facebook.com'	59.06
'blackplanet.com'	21.94
'vkontakte.ru'	5.61
'tagged.com'	4.95
'orkut.com'	3.13
'myspace.com'	2.36
'mixi.jp'	1.54
'hi5.com'	0.48
'adultfriendfinder.com'	0.47
'badoo.com'	0.46

(b) Social Networking Sites

Table 1: Analysis of most accessed domains

4.2 Fileformats

Among all the collected HTTP GET requests, `.html` was predominant with almost 54 % of all the requests. Around 32 % were caused by image formats, followed by JavaScript with 4.25 % of all GET requests. The details of the top 10 requested file extensions are shown in table 2.

Fileformat		
Extension	Description	Per cent (%)
'html'	HyperText Markup Language	53.83
'jpg'	JPEG image	18.15
'gif'	Graphics Interchange Format (GIF)	11.43
'js'	JavaScript	4.25
'css'	Cascading Style Sheets (CSS)	3.03
'png'	Portable Network Graphics	2.81
'xml'	Extensible Markup Language	2.62
'ico'	ICO image	0.77
'swf'	Shockwave Flash file	0.48
'rar'	RAR archive file format	0.20

Table 2: Top 10 file formats

4.3 Web browser types

Web browsers submit a text string known as “user agent string” to servers with details on the client version and the operating system they are running on. When

looking at the browser user agent string, various browser and operating systems combinations were seen. The browser used to access websites through the Tor network plays a crucial role for the anonymity of the end-user. TorButton [16] is a plugin for the Mozilla Firefox browser developed by Mike Perry and Scott Squires, which makes it possible for the user to switch between using the Tor network and browsing the web directly. Even more important, it disables many types of active content which could be used to bypass Tor and thus defeat Tor's anonymity protecting methods. To begin with, we inspected which were the ten most common user agent strings. Table 3 shows the top 10 browser user agent strings within our experimental data.

Browser	
Full User Agent String	Per cent (%)
'Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.0.7) Gecko/2009021910 Firefox/3.0.7'	18.86
'-'	4.48
'Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9) Gecko/2008052906 Firefox/3.0'	2.71
'Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.16) Gecko/20080702 Firefox/2.0.0.16'	1.81
'Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)'	1.66
'Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)'	1.64
'Mozilla/5.0 (Windows; U; Windows NT 6.0; en-US; rv:1.9) Gecko/2008052906 Firefox/3.0'	1.59
'Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)'	1.50
'Mozilla/5.0'	1.34
'Opera/9.63 (Windows NT 5.1; U; en) Presto/2.1.1'	1.31

Table 3: Top10 Browser (Raw user agent string)

TorButton uses a more constant user agent string in contrast to Firefox since the Firefox user agent string changes with every update of Firefox. This increases the anonymity set, as long as the used string is plausible and the used version is still in widespread use. By an analysis of the TorButton source code we identified nine distinct user agent strings that have been used by the different versions of TorButton. We found that solely 22 % of all the user agent strings matched one of the nine user agent strings used by the TorButton Firefox extension. Hence we argue that at least 78 % per cent of all traffic originated from a browser without the TorButton extension. It remains unclear if the users take additional preventive measure when using other browsers, however it seems unlikely that the majority of the non-TorButton users enforced all required browserside anonymity protection measures. Table 4 shows the Top 10 user agent strings, whether the user agent strings were used by TorButton and the revision of the TorButton source code. Furthermore the table shows the user agent strings and operating systems versions in a more readable format.

5 Further Tor Exit Server Attack Scenarios

A malicious exit server has many different options to gain knowledge of the Tor users, active as well as passive. Many HTTP requests leak information of

Browser		
Version	OS	Per cent (%)
TorButton > svn:r18954 (Firefox/3.0.7)	-	18.86
-	-	4.48
Firefox/3.0 (Gecko/2008052906), en-US	Windows XP	2.71
TorButton > svn:r16066 (Firefox/2.0.0.16)	-	1.81
Internet Explorer 6.0 SV 1.0	Windows XP	1.66
Internet Explorer 6.0	Windows XP	1.64
Firefox/3.0 (Gecko/2008052906)	Windows Vista	1.59
Internet Explorer 6.0	Windows 2000	1.50
Mozilla/5.0	-	1.34
Opera 9.63 (Presto/2.1.1), en	Windows XP	1.31

Table 4: Top 10 Browser (interpretation)

various kinds. Additional information can be gained by active content insertions and modifications.

5.1 Information leakage of sensitive information

Many HTTP requests leak information, sometimes even sensitive information. Even if the malicious exit server is unable to identify the originator of the request, it is able to gain knowledge only by watching the requests passively.

- *Search query strings:* Search queries are often submitted to a search engine via a HTTP GET request. If a Tor user searches information about e.g. a special disease, location information about a hotel, how to get to a certain place, or a recent political event, this gives hints about the identity, location or nationality of the user. Additional information can be deduced by language, browser and operating system to further reduce the anonymity set. This theoretical risk has also become evident in practice by the incident with AOL's release of search queries [17].
- *Social networks:* As described above social networking sites accounted for more than 7 per cent of the all the HTTP traffic captured by our Tor exit server. In the case of Facebook as well as other popular SNSs, HTTP requests include the user-id in plaintext. Because users often represent their realworld persona, SNSs users can easily be identified. The social graph of a user could furthermore reveal the identity of a SNS user. Thus an analysis of a user's id and corresponding social graph could completely deanonymize the Tor user. This is especially dangerous as many Tor users apparently use social networks.
- *Localization information* sent by a misconfigured browser reduces the anonymity set considerably. TorButton normalizes all user to use "en-us" within the browser user agent. This increases the size of the anonymity set for users

from the US, however decreases the size for the rest of the world. Other localization information can be retrieved from toplevel domains as we suspect that many users e.g. browse to their localized version of the Google search engine (google.ca, google.dk, ...) instead of the normalized google.com.
- *Other* sensitive and possibly incriminating content will be transmitted without proper usage of Tor, e.g. authentication cookies. However, as these are not transmitted in the HTTP GET request, we did not include them in our analysis. A motivated attacker surely will harvest those information as well, thereby possibly increasing the chance of defeating anonymity of a certain Tor users.

As an example we will use a search query on google maps, asking for driving instructions from Times Square to Central Park in New York. As it can be seen, plenty of parameters are part of the request, and of course the exact address coordinates as well:

```
http://maps.google.com/maps?f=d&source=s_d&saddr=times+square ->
  &daddr=central+park&hl=en&geocode=&mra=ls ->
  &sll=40.771133,-73.974187&sspn=0.053106,0.111494 ->
  &g=central+park&ie=UTF8&t=h&z=16
```

5.2 Script injection

Dynamic website content such as e.g. AJAX or Flash is hard to filter: either all dynamic content is blocked which results in poor usability, or dynamic content is allowed which opens the door for exit servers injecting malicious scripts. It has already been shown that the insertion of invisible iframes and Flash can defeat anonymity [18]. By injecting JavaScript within an invisible iframe it was possible to further reduce the requirements on the client to become vulnerable to this attack [19]. The authors even showed that their attack is feasibly by modifying the HTML meta refresh tag of a website, so without JavaScript.

By manipulating HTTP communication content, phishing attacks become possible if no communication encryption or verification is used. Instead of sending the data to the destination, a malicious exit server might save the data and present an error page.

5.3 File replacement

There exist many file formats commonly used on the Internet which could be used by an adversary to execute malicious code as the corresponding software handling the files has well known weaknesses. Among all the HTTP requests we have seen, up to 97993 requests, or 1% of the total requests were for files with known vulnerabilities and publicly available exploits. As we did not inspect the communication content and the transferred files itself, it remains unclear if the files could have been used for infecting the client or not. Instead, it can be seen as an upper bound for a new possible and yet not documented infection vector.

- executable (5590 requests): `.exe` files could get replaced or malicious code could be appended to the executable. This could result in remote code execution on the client side and could among other things be used to reveal the users identity.
- PDF (1619 requests): vulnerabilities in Adobe Reader and Acrobat as well as alternative PDF viewers could be used to execute malicious code in `.pdf` files.
- Office (400 requests): Microsoft Office has many documented vulnerabilites. The requests we monitored were for `.doc`, `.xls` and `.ppt` files.
- Mediafiles (23821 requests): Several Mediaplayer like e.g. Apples Quicktime, the Windows Media Player or VLC have documented flaws that are exploitable by manipulated media files. We encountered various requests that could be used for exploit piggybacking: `.avi` (10105 requests), `.wmv` (6616 requests), `.mp3` (4939 requests) or `.mpg` (1138 requests).
- other file formats (66563 requests): compressed file archives like `.zip` (9937 requests) and `.rar` (16512 requests) might get used by an attacker as they can be used to exploit vulnerabilities in the software itself; but also to add, manipulate or replace files within those archives. Shockwave flash files `.swf` (40114 requests) account for a major proportion of vulnerable file formats and could be used for client infection as well.

These vulnerabilities could be used to massively compromise anonymity on a large scale by exploiting vulnerable software, using Tor for cheap "man in the middle" attacks and even creating a botnet of Tor clients. However, it has to be noted that this is not a flaw of Tor but of rather an inappropriate use of it. Clients should verify the integrity of their files when using unencrypted communication.

5.4 Countermeasures and discussion

It is hard if not even impossible to prevent information leakage in HTTP requests, as the requests are often transmitting information of a certain user's context. The collection and aggregation of webservice specific information is non-trivial, but we showed that a great amount of information can be gathered by a passive adversary. In the following we briefly outline three methods that can help to mitigate security and privacy risks caused by a malicious Tor exit server:

Detection of malicious exit servers The most straightforward solution would be to detect bad exit servers and ban them from the Tor exit server list. McCoy et al. [3] proposed to use reverse DNS lookups in order to detect exit servers that run packet analyzer software with a host resolution feature. A complete passively adversary is almost undetectable. TorFlow is an application developed by Mike Perry [20] which supports exit server integrity checks. Hereby the basic principle is that: a reference file is used as a sample and downloaded through different Tor exit servers, cryptographic checksums are used afterwards

to check if a manipulation occurred. The basic idea works fine on files like binaries or static websites, dynamic content however is much harder to verify. For this reason, dynamic content is blocked by TorButton instead of analyzed for maliciousness.

User education The Tor network offers reliable anonymity in case if properly used. Awareness-campaigns as well as end-user education can help to ensure that people use Tor always in combination with TorButton as well as take special care of which services they use through Tor. The incident with the embassy mail accounts [6] has shown what might happen if Tor is used incorrectly, even seriously harming privacy instead of preventing de-anonymization and obfuscating user activity. Active content (e.g. Macromedia Flash) of all kind should be avoided if possible, and using trusted exit nodes could further reduce the risk of data leakage.

HTTPS The only solid protection of user data would be the use of strong encryption such as secure HTTP (HTTPS). The usage of the HTTPS protocol is unfortunately not always possible as many website operators do not support it, e.g. https://www.google.com redirects the user to http://www.google.com. At the time of writing the great majority of social networking providers fail to support HTTPS.

6 Summary and Conclusion

By collecting 9 x 10^6 HTTP requests we observed that up to 78 % of the Tor users browse the Internet without TorButton, the recommended software by the Tor community. The majority of users is therefore possibly vulnerable to sophisticated deanonymization attacks by an exit server, e.g. by injecting invisible iframes or scripts. 1 % of the requests were for vulnerable file formats, which could be used for exploit piggybacking. Even if the adversary running an exit server is completely passive, without modifying or redirecting communication content, an uncountable amount of sensitive information like search queries or authentication cookies are leaked. Furthermore, 7 % of all analysed HTTP connections were created by social networking services which leak plenty of personal information. To protect the Tor users, various tools like TorButton or TorFlow have been proposed and implemented. However, the only effective countermeasure at the time of writing is the use of end-to-end cryptography, namely HTTPS.

Future research in this area can be done to quantify the amount of sensitive information observable by an passive adversary. It has to take into account the different requirements for anonymization and/or censorship circumvention by the users. Additional research is needed on protection measures.

References

1. R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *13th USENIX Security Symposium*, San Diego, CA, USA, August 2004.
2. Tor: anonymity online. `https://www.torproject.org/`.
3. Damon McCoy, Kevin Bauer, Dirk Grunwald, Tadayoshi Kohno, and Douglas Sicker. Shining Light in Dark Places: Understanding the Tor Network. In *Proceedings of the 8th Privacy Enhancing Technologies Symposium (PETS 2008)*, Leuven, Belgium, July 2008.
4. Tor Project. Measuring Tor and Iran, 2009. `https://blog.torproject.org/blog/measuring-tor-and-iran`.
5. Forbes. PluggedIn: Web tools help protect human rights activists, 2009. `http://www.forbes.com/feeds/afx/2009/08/19/afx6794971.html`.
6. Dan Egerstad. DEranged gives you 100 passwords to Governments & Embassies. online, 2007. [Retrieved 2009-12-20].
7. J.F. Raymond. Traffic analysis: Protocols, attacks, design issues, and open problems. *Lecture Notes in Computer Science*, pages 10–29, 2001.
8. N. Mathewson and R. Dingledine. Practical traffic analysis: Extending and resisting statistical disclosure. *Lecture Notes in Computer Science*, 3424:17–34, 2005.
9. N. Hopper, E.Y. Vasserman, and E. Chan-Tin. How much anonymity does network latency leak? In *Proceedings of the 14th ACM conference on Computer and communications security*, page 91. ACM, 2007.
10. Steven J. Murdoch and George Danezis. Low-Cost Traffic Analysis of Tor. In *SP '05: Proceedings of the 2005 IEEE Symposium on Security and Privacy*, pages 183–195, Washington, DC, USA, 2005. IEEE Computer Society.
11. 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)*, Washington, DC, USA, October 2007.
12. S.J. Murdoch. Hot or not: Revealing hidden services by their clock skew. In *Proceedings of the 13th ACM conference on Computer and communications security*, page 36. ACM, 2006.
13. Lasse Øverlier and Paul Syverson. Locating hidden servers. In *Proceedings of the 2006 IEEE Symposium on Security and Privacy*. IEEE CS, May 2006.
14. dsniff. `http://www.monkey.org/~dugsong/dsniff/`.
15. apachelog. `http://code.google.com/p/apachelog/`.
16. M. Perry and S. Squires. Torbutton. `https://www.torproject.org/torbutton/`.
17. M. Barbaro and T. Zeller. A face is exposed for AOL searcher no. 4417749. *New York Times*, 9:2008, 2006.
18. Andrew Christensen. Practical Onion Hacking: finding the real address of Tor clients, 2006. `http://www.fortconsult.net/images/pdf/Practical_Onion_Hacking.pdf`.
19. T.G. Abbott, K.J. Lai, M.R. Lieberman, and E.C. Price. Browser-Based Attacks on Tor. *Lecture Notes in Computer Science*, 4776:184, 2007.
20. Mike Perry. Torflow: Tor network analysis. In *HotPETs 2009: 9th Privacy Enhancing Technologies Symposium*, page 14, Aug 2009.