ExitSniffer: Towards Comprehensive Security Analysis of Anomalous Binding Relationship of Exit Routers

Qingfeng Zhang^{1,2}, Xuebin Wang^{1,2(✉)}, Jinqiao Shi³, Meiqi Wang^{1,2},
Yue Gao^{1,2}, and Can Zhao^{1,2}

¹ Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
wangxuebin@iie.ac.cn

² School of Cyber Security, University of Chinese Academy of Sciences,
Beijing, China

³ School of Cyberspace Security, Beijing University of Posts
and Telecommunications, Beijing, China

Abstract. Tor exit relays are operated by volunteers and the trustworthiness of Tor exit relays need to be revisited in a long-term manner. In this paper, we monitored the Tor network by developing a fast and distributed exit relay scanner (ExitSniffer) to probe all exit relays over a period of 16 months continuously, seeking to expose the anomalous binding relationship phenomena of exit routers simply by comparing the returnIP and consensusIP. We totally find 1983 malicious exit relays which average contribute 10.12% bandwidth of total Tor exit relays bandwidth monthly, resulting tremendous threaten for Tor user's anonymity according to the current path-relay selecting algorithm. There exists two types of anomalous binding relationship consists 35 exit relay families, with different size ranging from 2 to 230, which are neither announced in the consensus document or detected by the Tor network.

Keywords: Malicious exit nodes · Exit sniffer · Tor network

1 Introduction

Tor provides a safe and concealed channel for clients to access the clear network. As of August 2021, there are approximately 1,300 exit relay nodes in the Tor network to provide services for clients [1]. Since the exit relay node is the last hop from the Tor network to the target website [2], it means that Tor's onion encryption algorithm will not be able to protect the traffic between the exit relay node and the target website. This may directly expose some client traffic without encryption means (such as HTTP requests) [3,4] to the attacker's vision. Therefore, the identity of the exit relay node is very sensitive, which is directly related to the security and privacy of client traffic.

Since the relay nodes of Tor are contributed by volunteers, attackers can implant malicious nodes with different roles into tor network at a very small

cost to realize their attacks, such as sybil attack [7]. A lot of work has proved that attackers have caused great harm to the anonymity and stability of tor network by manipulating controlled nodes [6–8]. Because the exit node acts as the middleman between the client and the target website, it is common for attackers to attack by controlling the exit node. [12], The reason why attackers are happy to implant malicious exit nodes may be that some government or school researchers try to destroy the anonymity of tor network; Another kind of attacker is for benefit. The attacker can hijack the transaction traffic and fake the content in the traffic by implementing man in the middle attack [8], so as to gain profits.

Tor officials and some researchers have launched defense against some common exit node attacks, such as traffic sniffing, DNS pollution and SSL based attacks [9, 19] through practical work. For example, Tor officials launched Tor Metric project [10] to measure the ecology in Tor network. Philipp Winter et al. designed the ExitMap tool to detect malicious exit nodes [9]. In 2016, Philipp Winter’s team again designed Sybilhunter tool to detect the sybil nodes of the Tor network [11]. These works have achieved remarkable results, but in the face of some new malicious exit node attacks, the results of the above methods are not satisfactory. When we observed the behavior of exit nodes in tor network for a long time, we find a phenomenon: the host IP address directly connected to the target website is inconsistent with the consensus IP address of exit nodes. We define it as a new attack mode of exit nodes. Unfortunately, the existing exit node probing tools can not detect this malicious behavior, because this silent attack method cannot trigger the detection module of the detection tool.

We have developed ExitSniffer system based on Python language, which depends on Ubuntu 16.04 system environment. ExitSniffer is designed with the idea of distributed clustering. It consists of three cloud hosts (with 1 core CPU and 4 GB of RAM) distributed in different countries, and can control multiple Tor clients to complete operations at the same time. The frequency of scanning nodes can be controlled by setting the timing module of ExitSniffer.

The principle of ExitSniffer is: use the exit node to be tested to access the website with the function of detecting IP through the Stem control protocol. It will compare the IP address returned by the website with the IP address of the exit node in the consensus file to judge whether the exit node is malicious.

By analyzing the results returned by exitsniffer, we find that the consensus IP of the malicious exit node has a malicious binding relationship with the IP returned by the website. Another phenomenon is that the IP returned by the website is in the same/24 network segment as the consensus IP of the exit node. However, we cannot distinguish whether the actual IP and consensus IP are different network cards of the same host, or different hosts. However, this malicious binding relationship indicates that there is a co-owner relationship between exit nodes.

To sum up, we make the following key contributions:

- We find that the exit node is not directly connected to the target website, and defined this type of exit node as: a malicious exit node with a proxy.
- We designed and implemented ExitSniffer, which is capable of detecting malicious exit node with proxies in consensus files in real time. It adopts distributed design, and is lightweight and easy to deploy.

- We analyze the influence of these malicious exit nodes with proxies on Tor networks and explore the co-owner relationship of these nodes.

2 Related Work

Due to the low threshold of implanting nodes into tor network, it promotes the continuous growth of tor network, but it will also make tor network absorb some malicious nodes. Up to now, tor network has been subject to Traffic Association attacks and node attacks by attackers, such as traffic confirmation attack, DoS attack, sybil attack, etc. However, some researchers have always supported the development of tor, and the communities involved in tor have been active. They have developed many tools to defend against tor attacks through their practical work.

Murdoch et al. [22] verified the IX level attack. By sampling the real traffic and extracting the statistical characteristics such as message sending rate and message length, the entity communication relationship was confirmed. Nasr et al. [23] Designed a traffic association system deepcorr, which uses the deep learning architecture to learn the traffic association function suitable for tor complex networks, and can connect Tor. Ling et al. [24] control the sending law of Tor cell on the controllable OR node. Wang et al. [25] proposed an attack scheme. Once the attacker finds that the target page of his exit node responds to traffic, he can inject malicious network links with empty images to make the browser on the client side download these links to generate specific traffic model.

However, there are some researchers who have been working on the defense of Tor. Akhoondi et al. [26] designed an efficient algorithm to judge whether the tor link can be associated by as traffic, and designed a new Tor client LASTor based on this algorithm. LASTor can avoid establishing tor links that can be associated by as traffic, so as to improve the security of tor dark network. Philipp Winter et al. designed the ExitMap tool to detect malicious exit nodes [9]. In 2016, Philipp Winter's team again designed Sybilhunter tool to detect the sybil nodes of the Tor network [11]. Sybilhunter integrates the functions of exitmap tool and adds HTML and HTTP injection detection.

By observing the exit nodes in the Tor consensus document for a long time, we find a malicious exit node with a proxy (MENP). However, the existing node scanning tools ignore this malicious node. Therefore, we designed a special software ExitSniffer to detect MENP nodes, which can scan all exit nodes in the consensus file in a short time. We hope that the ExitSniffer tool can become an extension of ExitMap and other tools to make up for the defense mechanism of malicious exit nodes. Next we will introduce the composition of ExitSniffer and analyze the behavior of MENP nodes.

Fig. 1. It shows the schematic diagram of ExitSniffer. With a distributed design, ExitSniffer can start multiple Tor clients at the same time, create Tor links through Stem libraries, visit websites with IP detection function, and finally output the results to the specified database.

3 The Design of ExitSniffer and Phenomenon

3.1 The Design of ExitSniffer

Before using exitsnifer, it is necessary to introduce its working principle. As shown in Fig. 1, ExitSniffer will download the exit node information in the latest consensus file every hour, use Stem control protocol [14] to create the circuits, judge whether the exit node is evil by using the target exit node to access the website with IP query function, and finally output the result to the specified database. Stem is a Python controller library for Tor. With it we can use Tor's control protocol to script against the Tor process, or build things such as Nyx. We use the Stem library to control Tor's circuit creation process. For example, we can decide when to establish a circuit and which exit node to choose as the last hop of the circuit. The target website we use also has special functions. It can return the IP of the host directly connected to it. We can deploy a website with this function ourselves, because it is more secure. But using public websites(such as <https://jsonip.com/>) is more labor-saving, as long as we would like to take some security risks. Generally speaking, the exit node is directly connected to the target website. Therefore, the IP of the exit node obtained by the target website should be consistent with the IP of the exit node in the consensus document. If this condition is not established, we believe that there is a problem with the exit node.

ExitSniffer is a distributed sniffer tool developed in Python can that detect malicious exit nodes with proxies. We use a distributed setup, utilizing 3 Virtual Machines (VMs) on cloud environment provided by Vultr¹. These virtual

¹ <http://vultr.com>.

machines are located in different countries including Singapore, France and the United States, so as to ensure the diversity of traces. Each VM is configured with 1 core CPU and 4 GB of RAM. On each VM, 10 docker instances are deployed, and each docker with a separate Tor process (version 0.4.4.6). Next, we will introduce the data set captured using exitsniffer.

3.2 Dataset

We set the detection frequency of ExitSniffer to be executed every two hours. A total of 7125,133 data records were collected from 2020-02-18 to 2021-08-18 by ExitSniffer system. Unfortunately, due to machine failure and other reasons, our data collection process was interrupted during the two months from 2021-02-18 to 2021-04-18, but this did not affect the following work. This dataset has 10 fields:

- consensus-IP:IP address of the exit node in the consensus file.
- actual-IP:ExitSniffer actually obtains the IP address of the exit node.
- bandwidth:Bandwidth of the exit node in the consensus file.
- flags:The labels of the exit node in the consensus file.
- or-port:The OR port of the exit node in the consensus file.
- fingerprint:The fingerprint of exit node in consensus file.
- nickname:The nickname of the exit node in the consensus file.
- spent-time:The time it takes the ExitSniffer to scan an exit node.
- status:ExitSniffer detects the result of an exit node.
- recording-time:The local time when the ExitSniffer scans an exit node.

It is worth noting that the data set mainly contains the information of exit nodes in the consensus file, so it depends on the accuracy of the consensus file.

4 Experimental Analysis

By testing exit node behavior for a long period of time, we find a hidden phenomenon: the IP address in the consensus document of the exit node is inconsistent with the IP address connected to the target website. This is similar to adding an proxy between the exit node and the target website. We believe that this behavior is malicious and mark the exit node with this behavior as MENP(a malicious exit node with a proxy). The inconsistency between the consensus IP address and the actual IP address may be caused by malicious operations of the trunk owner, such as man-in-the-middle attacks. Figure 2 shows that MENP nodes can reroute traffic to Tor network or route it to hosts outside the Tor network. By analyzing the IP actually returned by ExitSniffer, we find that some of MENP nodes route the client traffic to nodes outside the Tor network. Furthermore, the attacker may control multiple malicious exit nodes, and first aggregate the traffic relayed through these controlled exits to a controlled node outside the Tor network, and then route it to the client's target website. The behavior has a

Fig. 2. MENP nodes can decide whether to route the traffic that flows through them into or out of the Tor network. In Fig. 2(a), MENP nodes route traffic into the Tor network, and traffic eventually arrives at the target website from a random exit node. Figure 2(b) shows the MENP node routing traffic to a conspiracy host outside the Tor network

negative impact on the privacy protection of the client, because we don't know what happened between malicious exit node and the target website.

For malicious exit nodes with proxy (MENP) that route traffic outside the Tor network, there are two cases for Actual exit node IP (actual-IP) scanned by ExitSniffer:

- 1) Actual-IP is another IP address in the same/24 network segment of the Consensus IP address (Consensus IP) of the exit node. For example, by searching the Consensus file, the IP address of the exit node is 109.70.100.9, while the IP address detected by ExitSniffer is 109.70.100.27. It is impossible to tell whether such a node is a malicious one, even if its consensus IP is different from the one detected by the ExitSniffer, because there may be multiple network cards on the machine.
- 2) Actual-IP and the consensus IP address of the exit node are not in the same/24 network segment. As shown in Fig. 2(b), the client traffic is routed by the malicious exit node to other relays that are not part of the Tor network, during which the attacker is likely to carry out a man-in-the-middle attack.

Perhaps MENP nodes do nothing but route the traffic between the client and the target website to the proxy, which may also be a kind of goodwill behavior.

According to the work of Zhao Zhang et al., some websites have implemented IP blocking on Tor exit nodes [13]. Adding a proxy outside the Tor network behind the exit node can circumvent a site’s blocking of Tor, but this is potentially risky because it could expose more traffic from clients and target sites to attackers.

4.1 The Size of the Malicious Exit Nodes

We analyze the data of MENP nodes (malicious exit node with a proxy) since 2020-02-18 to 2021-08-18 (where 2021-02-18 to 2021-04-18 data is missing). We totally find 1983 malicious exit relays. Figure 3 shows the proportion curve of the number of malicious exit relays to the total number of exit nodes in the consensus file. It can be seen that the proportion curve (green line) has relatively large fluctuations, which means that the MENP node has a large churn rate, that is, the attacker deploys the MXNO malicious node and implements malicious behavior in a short period of time. However, from 2020-07-18 to 2020-08-18, the number of MENP nodes accounted for 16% of all exit nodes, which has reached a very large scale.

Fig. 3. This figure shows the loss of MENP nodes over time. It can be clearly seen that many MENP nodes run for a short time, so the proportion curve (green line) fluctuates greatly. (Color figure online)

4.2 Bandwidth Ratio of MENP Nodes

Tor’s selection of exit nodes follows the routing selection algorithm [15]. The client randomly selects a node as the exit node from the relay node set that

meets the exit node label rules by using the bandwidth weighting algorithm [16]. The calculation method of $weight_{final}$ of each alternative node is shown in the formula:

$$weight_{final} = weight * bw_n = \begin{cases} I_n * (W_{db}) * W_d * bw_n, & n \in S_{guard,exit} \\ I_n * (W_{gb}) * W_g * bw_n, & n \in S_{guard} \\ I_n * (W_{eb}) * W_e * bw_n, & n \in S_{exit} \\ I_n * (W_{mb}) * W_m * bw_n, & else \end{cases} \quad (1)$$

$$I_n(x) = \begin{cases} x, & n \in S_{dir} \\ 1, & else \end{cases} \quad (2)$$

Where S is the set of all nodes in the consensus file, n is a node in the set S , and bw_n is the consensus bandwidth of node n . $S_{guard,exit}$ is a set of nodes with Guard and Exit labels in set S . S_{guard} is set of nodes with only Guard labels in set S . S_{exit} is set of nodes with only Exit labels in set S . W_{db} , W_{gb} , W_{eb} , and W_{mb} can be obtained from the consensus file. W_{db} is the bandwidth weight with Guard and Exit tags, W_{gb} is the bandwidth weight with Guard tags, W_{eb} is the bandwidth weight with Exit tags, W_{mb} is the bandwidth weight without Guard or Exit tags, W_d is the W_{ed} parameter in the consensus file, W_g is the W_{eg} parameter in the consensus file, W_e is the W_{ee} parameter in the consensus file, and W_m is the W_{me} parameter in the consensus file.

Query parameters related to consensus files and make standardized discovery:

$$W_g = W_{gb} = W_m = W_{mb} = W_e = W_{eb} = W_d = W_{db} = 1 \quad (3)$$

It can be obtained by combining Eq. 1 and Eq. 2:

$$weight_{final} = bw_n \quad (4)$$

According to the weighted bandwidth selection algorithm, the greater $weight_{final}$ of the node, the greater the probability of the node being selected. This means that the greater the bandwidth of the node, the greater the probability P_n of being selected by the client as the last hop of the circuit.

$$P_n = \frac{weight_{final,n}}{\sum_{i=0}^m weight_{final,i}} = \frac{bw_n}{\sum_{i=0}^m bw_i}, i \in (0, m) \quad (5)$$

In the above formula, P_n is the probability that the client selects exit node n , bw_i is the consensus bandwidth of exit node i , and m is the number of exit nodes in the consensus file.

Algorithm 1 is the weighted bandwidth algorithm of the exit node, and Algorithm 2 shows the algorithm for selecting the index of each node according to the weighted bandwidth value of each node. Therefore, we can deduce Eq. 1 and Eq. 5.

In Algorithm 2, even if the index of the selected node is found, the loop will not jump out immediately. This is to resist the attack of time consumption, and the attacker cannot calculate the value of *rand_value* through the running time of the algorithm.

Algorithm 1. Exit node compute weighted bandwidth

Input: all of relays in consensus file S

Output: the list of weighted bandwidths for exit nodes Bw_out

```

1:  $Bw\_out \leftarrow$  an empty list
2:  $We \leftarrow \text{get\_bw\_weight}(\text{"Wee"}) / 10000$ 
3:  $Wm \leftarrow \text{get\_bw\_weight}(\text{"Wem"}) / 10000$ 
4:  $Wd \leftarrow \text{get\_bw\_weight}(\text{"Wed"}) / 10000$ 
5:  $Wg \leftarrow \text{get\_bw\_weight}(\text{"Weg"}) / 10000$ 
6:  $Wgb \leftarrow \text{get\_bw\_weight}(\text{"Wgb"}) / 10000$ 
7:  $Wmb \leftarrow \text{get\_bw\_weight}(\text{"Wmb"}) / 10000$ 
8:  $Web \leftarrow \text{get\_bw\_weight}(\text{"Web"}) / 10000$ 
9:  $Wdb \leftarrow \text{get\_bw\_weight}(\text{"Wdb"}) / 10000$ 
10: for each  $n \in S$  do
11:   if  $n$  is guard and  $n$  is exit then
12:      $weight \leftarrow (\text{is dir ? } Wdb * Wd : Wd)$ 
13:   else if  $n$  is guard then
14:      $weight \leftarrow (\text{is dir ? } Wgb * Wg : Wg)$ 
15:   else if  $n$  is exit then
16:      $weight \leftarrow (\text{is dir ? } Web * We : We)$ 
17:   else
18:      $weight \leftarrow (\text{is dir ? } Wmb * Wm : Wm)$ 
19:   end if
20:    $weight_{final} \leftarrow weight * bw\_n$ 
21:    $Bw\_out.add(weight_{final})$ 
22: end for

```

Algorithm 2. Choose an exit node by weight

Input: the list of weighted bandwidths for exit nodes Bw_out

Output: the index of the selected exit node idx

```

1:  $total \leftarrow \text{sum}(Bw\_out)$ 
2:  $rand\_value \leftarrow \text{randint}(0, total)$ 
3:  $idx \leftarrow -1$ 
4:  $total\_so\_far \leftarrow 0$ 
5: for each  $bw \in Bw\_out$  do
6:   if  $total\_so\_far > rand\_value$  then
7:      $idx \leftarrow Bw\_out.index(bw)$ 
8:   else
9:      $total\_so\_far \leftarrow total\_so\_far + bw$ 
10:   end if
11: end for

```

Fig. 4. It shows the bandwidth capabilities of MENP nodes.

Figure 4 shows that the bandwidth curve of the MENP node is basically stable. But it fluctuates greatly in the proportion of the total exit bandwidth. This is because the large churn rate at the Tor exit point causes the total export bandwidth to fluctuate greatly [11]. Through statistical analysis, the bandwidth of MENP nodes accounts for 10.12% of the total exit bandwidth. According to Eq. 5, it can be concluded that the probability of clients selecting these malicious exit nodes as the last hop is 10.12%. This is a very scary thing, malicious exit nodes listen to or hijack about 10.12% of the exit traffic without being detected by Tor officials.

4.3 Behavior Exploration of MENP Nodes

After 16 months of observation and data collection, We find that MENP nodes have two forms when routing client traffic: 1) rerouting traffic to Tor network; 2) routing traffic to hosts outside the Tor network.

Traffic Is Routed to the Tor Network (MRIT, Malicious Routing into the Tor Network). When using the ExitSniffer system to scan the exit nodes, we find that many exit nodes reroute client traffic to the Tor network. We suspect that the reasons for this are as follows: 1) The circuit is replaced. When Tor detects a circuit failure, it will select another circuit. Therefore, the exit node IP detected by exitsnifer is inconsistent with the target exit node’s consensus IP 2) Malicious operations by node owners. Some researchers reroute traffic to tor network in order to collect more traffic information.

Traffic Is Routed Outside the Tor Network (MROT, Malicious Routing Out of the Tor Network). The attacker manipulated the exit node to route the traffic of the client to the outside of Tor network. Although it is not ruled out that the exit node is innocent, it may be that the connection between the exit node and the target website is hijacked by the attacker. We cannot distinguish it, so we uniformly believe that the malicious exit node leads to such behavior.

Fig. 5. It shows the ratio of MRIT malicious nodes to MROT malicious nodes.

Figure 5 shows that the number of MROT malicious exit nodes accounts for 67.59% of the total number of MROT malicious exit nodes. Figure 6 shows the bandwidth capacity of MROT malicious exit nodes, which accounts for 68.30% of the total number of MROT malicious exit nodes. Therefore, we turn more attention to MROT malicious exit nodes. We were surprised to find that when ExitSniffer scanned the malicious exit node with MROT behavior, the actual IP address obtained was in the same/24 network segment as the consensus IP address of the exit node. We were inclined to think that the ExitSniffer system had errors in obtaining the real IP address. We obtained the IP addresses of other network cards under the host, because we assume that if two IP addresses are in the same/24 network segment. We will aggregate them into one IP address. Such data records are common in our data set. For example: Consensus IP is on the same/24 network segment as actual IP, and ACutal IP1 is on the same/24 network segment as actual IP2.

Figure 7 shows the comparison of the number of malicious nodes with MROT behavior on whether the actual returned IP and the consensus IP are in the same/24 network segment. Most of the malicious exit node consensus IP and the actual measured IP are not in the same subnet. Moreover, the consensus IP and the actual returned IP present a one-to-one, one-to-many, many-to-one,

Fig. 6. It demonstrates the bandwidth capabilities of MROT malicious nodes.

Fig. 7. This figure shows the number of malicious nodes with MROT behavior that actually return IP addresses versus consensus IP addresses in the same/24 network segment.

and many-to-many malicious binding relationship. Table 1 shows the one-to-one relationship between consensus IP and actual IP, and Fig. 8 shows an exit node may be bound to multiple hosts outside the Tor network. Figure 9 shows that multiple exit nodes are bound to one or more hosts outside the Tor network. Our intuition is that multiple exit nodes are bound to one or more nodes outside the Tor network, which means that these exit nodes have a “co-owner” relationship [17, 18], which we will discuss in the next section.

Table 1. Consensus IP and actual IP have a one-to-one mapping relationship.

Consensus IP	Actual IP
93.133.12.195	2.243.1.65
83.236.208.78	95.223.285.221
117.222.107.93	61.1.215.1
191.34.116.131	177.18.199.95
31.13.195.173	87.120.37.148
73.174.133.218	67.163.135.106
82.202.170.31	82.146.55.139
86.128.30.92	81.151.109.84
158.69.35.227	54.39.101.136

Fig. 8. It shows that a consensus-IP corresponds to multiple actual-IP.

4.4 The co-owner Relationship of the Malicious Exit Node

Multiple exit nodes route traffic to one or more IP addresses outside the Tor network, and we suspect that these exit nodes are held by the same person or organization, i.e., there is a “co-owner” relationship. We used graph algorithm and Gephi tool to display the results of family aggregation. A total of 35 co-host families were found, and the largest family included 230 exit nodes.

Table 2. Different actual-IPS are bound to the same group of consensus-IPS. It means that these actual-IPS can be aggregated after exceeding the overlap threshold. Finally, consensusIPs will form a large family.

Actual IP1	Actual IP2	The number of overlapping consensus IP addresses
109.70.100.32	109.70.100.33	79
109.70.100.32	89.31.57.5	66
109.70.100.32	104.218.63.119	43
109.70.100.33	195.206.107.147	17

more users' traffic to attackers, so it is reasonable for us to define this kind of node as malicious exit node.

5 Conclusion

In this paper, we revisited the trustworthiness of Tor exit relays. We designed and developed the ExitSniffer tool to continuously pay attention to scale and bandwidth of malicious exit relay nodes, the consensus IP is inconsistent with the actual returned IP, over a period of 16 months. By analyzing the anomalous binding relationship phenomena of malicious exit nodes, we totally find 1983 malicious exit relays which average contribute 10.12% bandwidth of total Tor exit relays bandwidth monthly, resulting tremendous threaten for Tor user's anonymity according to the current path-relay selecting algorithm. Besides, according to our results, there exists two types of anomalous binding relationship consists 35 exit relay families which are neither announced in the consensus document or detected by the Tor network.

Acknowledgements. This work is supported by the Key Research and Development Program for Guangdong Province under grant (No. 2019B010137003), Fundamental Research program of National Defence (JCKY2019211B001) and the Strategic Priority Research Program of Chinese Academy of Sciences (XDC02040400).

References

1. The number of exit nodes. <https://metrics.torproject.org/relayflags.html>. Accessed 18 Aug 2021
2. Dingledine, R., Mathewson, N., Syverson, P.: Tor: the second-generation onion router. In: USENIX Security. USENIX Association (2004)
3. Bernstein, D.J.: Curve25519: new Diffie-Hellman speed records. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 207–228. Springer, Heidelberg (2006). https://doi.org/10.1007/11745853_14
4. Huber, M., Mulazzani, M., Weippl, E.: Tor HTTP usage and information leakage. In: De Decker, B., Schaumüller-Bichl, I. (eds.) CMS 2010. LNCS, vol. 6109, pp. 245–255. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13241-4_22

5. Douceur, J.R.: The Sybil attack. In: Druschel, P., Kaashoek, F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429, pp. 251–260. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45748-8_24
6. Marlinspike, M.: Sslstrip. <http://www.thoughtcrime.org/software/sslstrip/>
7. Ryan, P.: Security expert used tor to collect government e-mail passwords (2007)
8. TOR exit-node doing MITM attacks. <http://www.teamfurry.com/wordpress/2007/11/20/tor-exit-node-doing-mitm-attacks/>
9. Winter, P., et al.: Spoiled onions: exposing malicious tor exit relays. In: De Cristofaro, E., Murdoch, S.J. (eds.) PETS 2014. LNCS, vol. 8555, pp. 304–331. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08506-7_16
10. The Tor Project. Tor Metrics. <https://metrics.torproject.org/>
11. Winter, P., et al.: Identifying and characterizing Sybils in the Tor network. ArXiv abs/1602.07787 (2016)
12. How Malicious Tor Relays are Exploiting Users in 2020 (2020). <https://nusenu.medium.com/how-malicious-tor-relays-are-exploiting-users-in-2020-part-i-1097575c0cac>
13. Zhang, Z., Zhou, W., Sherr, M.: Bypassing tor exit blocking with exit bridge onion services. In: ACM Conference on Computer and Communications Security (CCS) (2020)
14. The Tor Project. Stem Docs (2013). <https://stem.torproject.org>
15. Hanley, H., Sun, Y., Wagh, S., Mittal, P.: DPSelect: a differential privacy based guard relay selection algorithm for tor. Proc. Priv. Enhancing Technol. **2**, 166–186 (2019). <https://doi.org/10.2478/popets/20190025>
16. The Tor Project. Relays with Exit, Fast, Guard, Stable, and HSDir flags (2013). <https://metrics.torproject.org/network.html>
17. Sanatinia, A., Noubir, G.: Honey onions: a framework for characterizing and identifying misbehaving Tor HSDirs. IEEE Conf. Commun. Netw. Secur. (CNS) **2016**, 127–135 (2016)
18. Chen, M., Wang, X., Shi, J., Yue, G., Gan, Z.: Towards comprehensive security analysis of family phenomenon of Tor hidden service. In: ICICS 2019 (2019)
19. Hodges, J., Jackson, C., Barth, A.: RFC 6797: HTTP Strict Transport Security (HSTS) (2012). <https://tools.ietf.org/html>
20. Alexa: The top 500 sites on the web (2013). <http://www.alexa.com/topsites>
21. Singh, R., et al.: Characterizing the nature and dynamics of tor exit blocking. In: USENIX Security Symposium (USENIX) (2017)
22. Murdoch, S.J., Zieliński, P.: Sampled traffic analysis by internet-exchange-level adversaries. In: Borisov, N., Golle, P. (eds.) PET 2007. LNCS, vol. 4776, pp. 167–183. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-75551-7_11
23. Nasr, M., Bahramali, A., Houmansadr, A.: Deepcorr: strong flow correlation attacks on Tor using deep learning. In: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, pp. 1962–1976 (2018)
24. Ling, Z., Luo, J., Yu, W., et al.: A new cell counter based attack against Tor. In: Proceedings of the 16th ACM conference on Computer and Communications Security, pp. 578–589 (2009)
25. Wang, X., Luo, J., Yang, M., et al.: A potential HTTP-based application-level attack against Tor. Futur. Gener. Comput. Syst. **27**(1), 67–77 (2011)
26. Akhoondi, M., Yu, C., Madhyastha, H.V.: LASTor: a low-latency AS-aware Tor client. In: 2012 IEEE Symposium on Security and Privacy, pp. 476–490. IEEE (2012)

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.