Deanonymizing Schemes of Hidden Services in Tor Network: A Survey

Sabita Nepal, Saurav Dahal and Seokjoo Shin
Department of Computer Engineering
Chosun University, Gwangju, South Korea
Email: sabitanepal2002@gmail.com, dahal.saurav@gmail.com and sjshin@chosun.ac.kr (corresponding author)

***Abstract*— Tor hidden services are used to provide a TCP based service to users while not exposing the hidden server's IP address so as to attain obscurity and anti-censorship. However, hidden services are abused in numerous ways. Hidden services are being misused for illegal services like pornography, drug trading information etc. For the sake of prevention and warning for misuse of Tor hidden services, studies on revealing hidden server's identities are somewhat inevitable. In this paper, we give an overview over the existing attacking schemes, their comparisons, explain their key ideas and show their interrelations. Comparison study gives a guideline of deanonymizing hidden services in terms of simplicity and effectiveness.**

***Keywords* – Tor network, Anonymity, Hidden services, Deanonymization.**

I. INTRODUCTION

Tor [1] is a low-latency anonymous communication system that supports TCP application over the internet, which gives obscurity services to the users such as web browsing, secure shell, and instant messaging. It provides users with anonymity service, helps to circumvent Internet censorship [14], and supports hidden services to preserve the anonymity of web services [2]. Tor presently supports multiple millions of users [12]. Due to progressively high demand of privacy protection, the Tor network has seen steady growth, in volunteer based Tor onion routers. Tor supports the hidden services to preserve the obscurity of these web services.

Hidden services are the websites located inside the Tor Networks, which receive inbound connection only through Tor. Tor is necessary to access hidden services [8]. Hidden services are accessed through its onion address, thus it lets user publish internet sites obscuring their IP address and location.

However hidden services are misused for illegal services like drug trading information [3], child pronography website [9] [10], illegal arms trading etc. The condition is severe if a hidden service hosts any of these websites because it blindly provides a protection to such illegal content which is hard to take down. There are numbers of black market exiting on Tor hidden services. Silk Road is one of them, which operates widely in smuggled goods using Bitcoin as currency.

Concerns about privacy and security have received greater attention with the rapid growth of the illegal activities within the Tor network. Researchers are concerned about the different attacking schemes to deanonymize the hidden service. So far there are many different attacking schemes which reveals the Tor hidden services. By revealing those hidden services, we can shutdown those hidden services which are hosting illegal sites. Thus, in this survey paper, we focus on existing attacking schemes, their working principal and comparisons between them. By comparing them, readers can understand the pros and cons of each scheme and thus they can apply the suitable scheme to deanonymize the hidden service.

The rest of the paper is organized as follows: In Section II, we introduce the components of Tor, its basic operations and the protocol of hidden service. In Section III, we present the three different existing attacking schemes, along with their comparison. The paper is concluded in Section IV.

II. TOR NETWORK

Tor is an application-level overlay network enabling anonymous communication between clients and arbitrary Internet destinations through onion routing. Clients make an anonymous communication to a server by tunneling their traffic through a chain of three Tor relays. In this section, we first introduce the Tor network and then present its basic operation and the protocol of hidden services.

A. Tor Overview

Tor is an overlay network for anonymous communication in which each onion router (OR) runs as a normal user-level process without any special privileges. It is a open source project and provides anonymity service for TCP applications [2]. Each OR maintains a TLS [11] connection to every other OR. Each user runs local software called an onion proxy (OP) to fetch directories, establish circuits across the network, and handle connections from user applications. These onion proxies accept TCP streams and multiplex them across the circuits. The OR on the other side of the circuit connects to the requested destinations and relays data. Figure 1 illustrates the basic architecture of Tor network. The following components are involved in the typical use of Tor network:

- *Tor clients*: A Tor client requests the data to be downloaded from the server. It installs a local software,

onion proxy (OP), in which application data are packed into equal-sized cells (512 bytes) and delivers them into Tor network. A cell is the basic transmission unit of Tor.

- *Onion routers* (OR): The ORs are the relays volunteered by different volunteers all over the world.
- *Directory servers*: Directory servers hold the information of ORs and hidden services, such as the public keys of routers and hidden servers.
- *Application servers*: It supports TCP applications such as a web service and an IRC service.

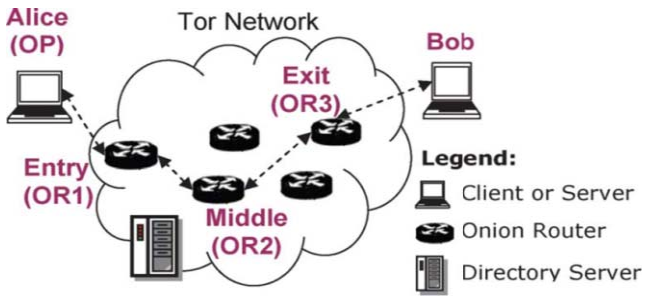

Figure 1. Tor network [4] .

B. *Hidden Services*

Tor provides anonymity to websites and other servers. Servers designed to receive inbound connections only through Tor are called hidden services. Instead of revealing a server's IP address, a hidden service is accessed through its onion address, which is the hash of its public key. Tor hidden service comprised of following important nodes:

- *Introduction Point* (IP): These are Tor relays which are chosen by hidden services and used to make connection with client.
- *Rendezvous Point* (RPO): These are chosen by the Tor Client and used to forward data between the client and the hidden server.
- *Directory Server* (DS): DS has the information about the Tor network nodes and these information are used to contact hidden services.
- *Hidden server* (HS): These are the server inside the Tor network which hosts the hidden services.

The hidden service design is based on connecting two circuits, one created by the client and other by HS, on a commonly agreed Tor relay. This Tor relay, thus referred to as a *rendezvous point,* which relays messages by forwarding outgoing messages from the client-side circuit to the server side circuit and vice versa. So as to shield RPOs from attacks, a hidden service picks a group of Tor relays as *introduction points* that work just like the RPOs. IPs are solely used for transferring one message containing the placement of the chosen RPO. So as to just accept client requests, the hidden service publishes a *hidden service descriptor* containing a signed list of IPs to DSs from where clients can download them [13].

A normal setup of communication between a client and the specific HS is shown in Figure 2. The procedure of this communication is as follows:

1. The HS first selects several ORs as IPs and builds the circuits to these IPs by sending the RELAY_COMMAND-_ESTABLISH_INTRO cell, and the IP replies with the RELAY_COMMAND_INTRO_ESTABLISHED cell to inform the HS that the circuit is established.
2. Once the circuits to IPs are established, the HS establishes a circuit to the DS and advertises the service descriptor to the DS, including the public key of the HS and the information regarding the IPs. Then the owner of the HS can post the onion address in a public place to attract users to access the hidden service via Tor.
3. When a Tor client obtains the onion address, the client creates a circuit to the DS and fetches the relevant information advertised by the hidden service. Then the client learns the IPs of the hidden service.
4. Now the client selects a RPO and builds a circuit to the RPO. The client will send a RELAY_COMMAND_ESTA-BLISH_RENDEZVOUS cell, which carries a rendezvous cookie, to the RPO, which replies with a RELAY_COM-MAND_RENDEZVOUS_ESTABLISHED cell to indicate the successful circuit establishment.
5. The client creates a three-hop circuit to one of the IPs and transmits a RELAY_COMMAND_INTRODUCE1 cell to the chosen IP. The cell carries the information such as the RPO, rendezvous cookie and the Diffie-Hellman data g^x generated by the Tor client.
6. Once the IP receives the RELAY_COMMAND_INTRO-DUCE1 cell, it replies with a RELAY_COMMAND-_INTRODUCE_ACK cell to the client. After the client receives this ACK cell, it tears down this circuit to the IP.
7. The IP repacks the RELAY_COMMAND_INTRODUCE1 cell into a RELAY_COMMAND_INTRODUCE2 cell, and then sends the RELAY_COMMAND_INTRODUCE2 cell to the HS. Once the HS receives this cell, it knows the information of the RPO, rendezvous cookie and Diffie-Hellman data g^x. The HS can generate the Diffie-Hellman data g^y and derive the key $K = g^{xy}$.
8. Then the HS builds a circuit to the RPO and sends a RELAY_COMMAND_RENDEZVOUS1 cell to the RPO via the circuit. When RPO obtains this cell, it compares the rendezvous cookie from the cell and the one from the Tor client. Once the rendezvous cookies are matched, the RPO removes the rendezvous cookie from the RELAY_COMMAND_ RENDEZVOUS1 cell and repacks the rest data into RELAY_COMMAND_RENDEZVOUS2 cell, and then forwards the cell to the client.
9. When the Tor client receives the RELAY_COMMAND-_RENDEZVOUS2 cell, it can generate the key $K = g^{xy}$ using g^y and verify it based on H(K). In this way, the client and HS complete the handshake. Then the client sends a RELAY_COMMAND_BEGIN cell to establish a stream to the HS via six hop circuit.
10. When the client send the RELAY_COMMAND_BEGIN cell to the HS in order to open a stream between the client and the HS, our controlled RPO detect this special cell based on the hidden service protocol [15].

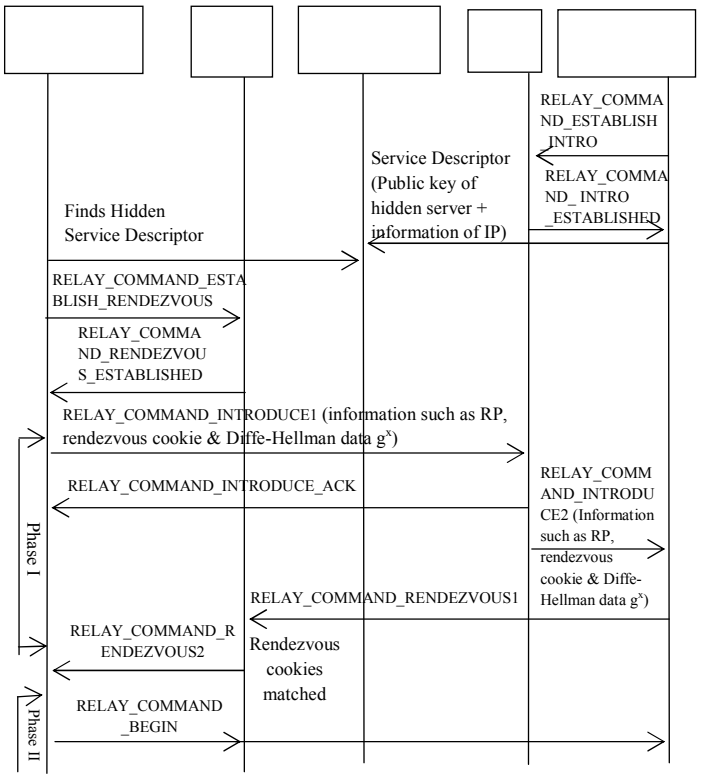

Figure 2. Procedure of establishing a connection between the Tor client and the specific hidden server.

III. ATTACKING SCHEMES

Hidden service offers a communicator obscurity for a persistent service. Users are able to connect to the service through Tor without knowing its location. These hidden services can host the illegal websites whose consequences can be severe. So these HSs need to be revealed. Deanonymizing hidden service is the strategy in which ip address of anonymous websites is revealed. One way to deanonymize these hidden services is by attacking them. Till now researchers have developed many different attacking schemes, which are discussed below.

In this section we discussed about the three different attacking schemes. In these attacking schemes there are some common features:

- All attacking schemes need adversary client and adversary guard node to deanonymize hidden service.
- In all attacking schemes, hidden services are forced to choose the compromised guard nodes as their entry nodes.
- All attacking schemes reveal the IP address of hidden service.

3-1. Manipulating Tor Cells Method

This method is based on a protocol-level discovery approach. Attacker controls a Tor client, a RPO, several entry ORs, and a central server. Through this method, it is easy to deploy the detection system. Compared to traffic analysis based methods [16], this approach is significantly faster, fully automatic and can quickly locate the HS using only several cells.

This approach is accurate with an observed detection rate of 100% and has an observed low false positive of 0% [5]. It works on the protocol level and is oblivious to traffic pattern i.e. it is more general and can be used to identify malicious hidden services.

In this method the HS discovery process is divided into three phases. In phase I, a client presumably identifies the HS as shown in Figure 2. After identifying a HS, in phase II, the HS is verified. At the end of the phase I, a client sends RELAY_COMMAND_BEGIN cell to the HS. As the RPO catches this cell, it modifies one bit of the cell and forwards it to the HS. Modification of a cell at RPO [5] is shown in Figure 3. The RPO also needs to send a timestamp of the manipulated cell to the central server. Due to the lack of integrity verification, other ORs cannot detect the manipulated cell.

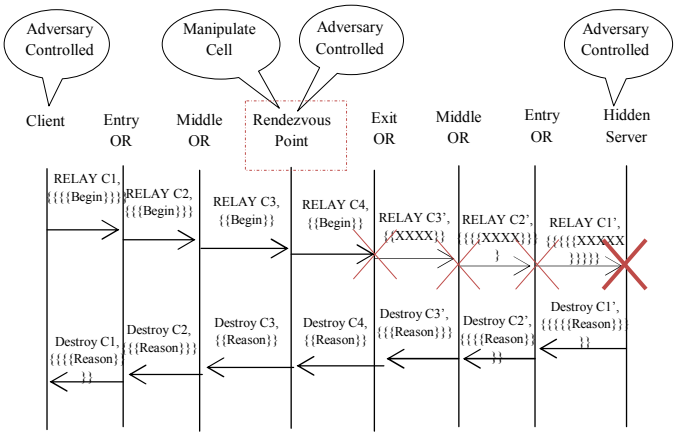

Figure 3. Modify a cell at the RPO

When the manipulated cell reached the HS, the HS cannot recognize this cell, and the circuit between the client and HS will tear down by sending a CELL_Destroy cell. Controlled entry OR first receives this cell, and it reports the cell type, the timestamp of the cell, circuit ID and the source IP address of the cell to the central server. Destroy cell reach the client through the RPO. At the RPO, it needs to report the timestamp of this cell to the central server. Finally by time correlation, we can find the IP address of HS.

In this method, hidden service should choose an adversary entry node as its guard node. For this to happen, the number of compromised entry nodes should be very high, almost 30% [5] of entire Tor entry routers, which is very expensive in terms of financial aspect.

3-2. Cell count based method

In this method, arbitrary Tor relays are anonymously and selectively disabled with very low cost to the attacker; this attack is efficient to the extent that an adversary interested in

censorship could disable, instead of blocking Tor by simply disabling all relays or intelligently targeting crucial subsets of relays, e.g., those providing high network throughput or authoritative directory services. This attack, also known as Sniper attack, is classified as a cell count attack since it deanonymizes a hidden service by counting the numbers of cells at an adversarial guard [6].

In this method, we deanonymize hidden services by selectively disabling relays through sniper attack, heavily influencing the paths to those in control of the adversary's. Thus this attack imposes real, significant threats to Tor's users, and it constitutes the most devastating attack against the Tor network to date.

A. Sniper Attack

The Sniper attack is performed in order to kill any arbitrary relay. Figure 6 shows the steps of this sniper attack as follows:

- A client creates a circuit using the target as the entry.
- A client requests data from a server and the server responds with data.
- A client stops reading the data from the target entry.
- A client periodically sends the SENDME packet to the server, but it doesn't read from the TCP connection to the target entry.
- The target entry buffers the data until the Tor process is terminated by the OS.

Since this method requires a target entry node of hidden service to be terminated by buffering all free memory, it could take a good amount of time for a system with big free memory.

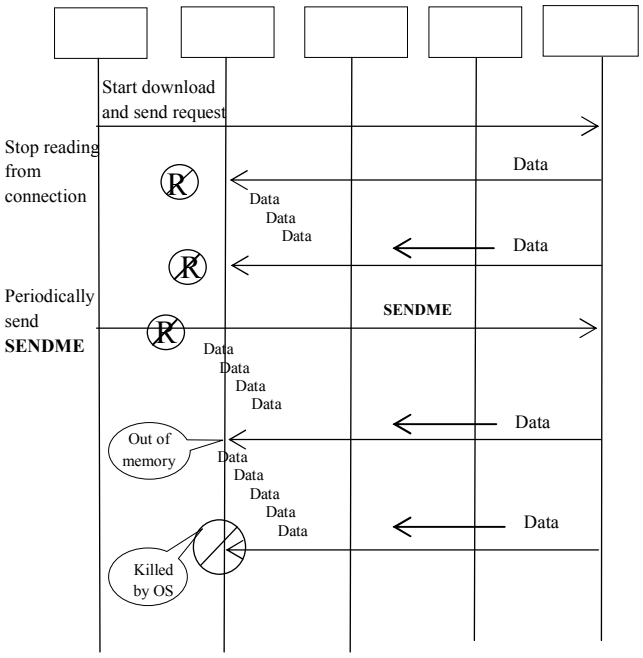

Figure 6. Procedure of the sniper attack

B. Deanonymizing Hidden services

The flow chart of the deanonymization process of the hidden services is shown in Figure 7.

- First of all, an adversary client makes HS to build new rendezvous circuits to learn its guard.
- Snipe HS guard to force reselection.
- Repeat until HS chooses an adversarial guard.

At first the adversary guard runs one guard relay (G_A). For

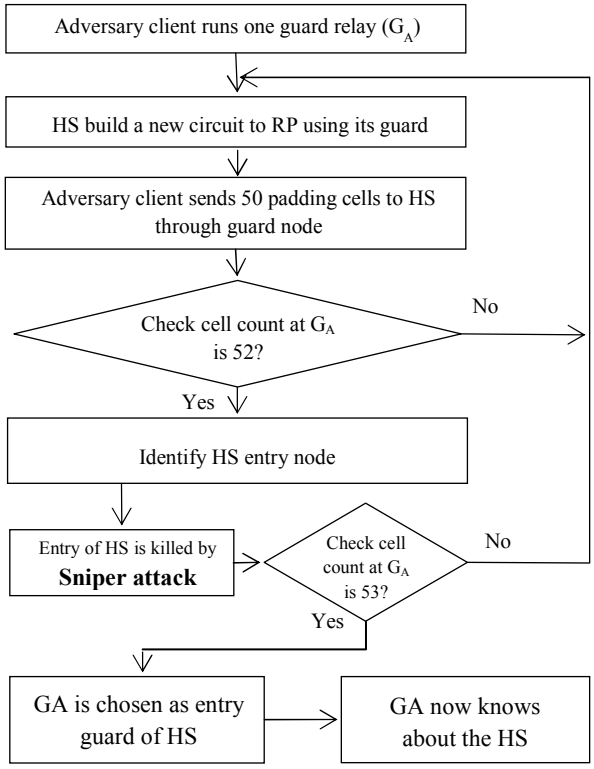

Figure 7. Flow chart of deanonymizing Hidden Services

deanonymization of hidden service, HS should choose one of adversarial relays as guard node. For this, HS builds a new circuit to RP. Adversary client sends 50 padding cells to the HS through G_A. If the cell counter at guard relay run by adversary client is 52 then the attacker can identify the entry node of the HS. Then, the entry node of the HS is sniped by sniper attack and HS is forced to reselect guard node. This process is repeated until an adversarial guard is chosen by HS. To check whether a compromised guard node is chosen by HS or not, an adversary client sends 50 padding cell to HS. If the cell counter at Adversary guard is 53 then the attacker can identify Hidden Service.

3-3. Padding Cell Method

In this attack, the hidden service is forced to establish rendezvous circuits to the RP controlled by the attacker. Upon receiving a RELAY_COMMAND_RENDEZVOUS1 cell with the attacker's cookie, the RP generates traffic with a special signature. This signature can be identified by the attacker's middle node. Note that a special PADDING cell

mechanism in Tor simplifies generation of a signature traffic, which is discarded at the recipient side, and is thus unnoticeable to the HS.

The steps of the attack is as in the Figure 8.

1. At first, the attacker sends a RELAY_COMMAND-_INTRODUCE1 cell to the IP of the HS indicating the address of the RPO.
2. The IP forwards the content in a RELAY-_COMMAND_INTRODUCE2 cell to the hidden service.
3. The HS establishes a three-hop circuit to the RPO by sending RELAY_COMMAND_RENDEZVOUS1 cell.
4. When the RPO controlled by attacker receives RELAY-_COMMAND_RENDEZVOUS1 cell, it sends 50 PADDING cells to the HS through the same path.
5. As HS receives this PADDING cell, these cells are silently dropped by HS.
6. RPO sends a DESTROY cell through the rendezvous circuit leading to the closure of the circuit.

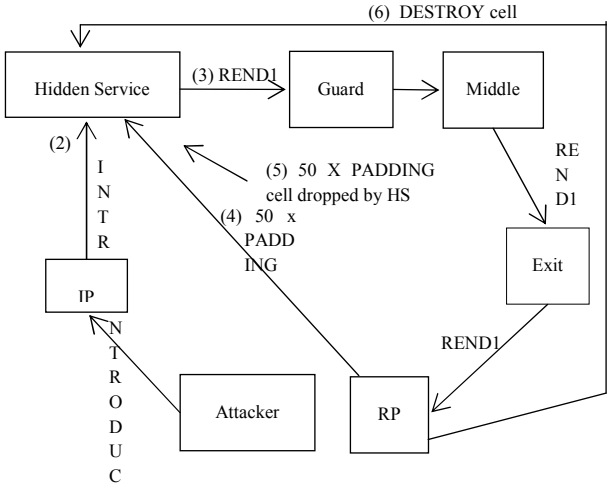

Figure 8: Revealing the guards

The attacker's guard node monitors traffics on the circuits passing through it. Whenever it receives a DESTROY cell over a circuit, it checks: (1) whether the cell was received just after the RPO received the RELAY_COMMAND_REN DEZVOUS1 cell, (2) the number of the forwarded cells: 3 cells up the circuit and 53 cells down the circuit. If these conditions are satisfied, then the attacker decides that her guard node has been chosen for the hidden service[7].

3-4. Comparision

Table 1 shows the comparisons of three different attacking schemes. In this comparison, the readers can see the pros and cons of different attacking schemes. In manipulating Tor cells method, the time required to deanonymize hidden service is very low in comparison to remaining other two schemes. But, the main cons of this scheme are that it requires a very large amount of compromised entry nodes which can be very expensive in financial aspect. Otherwise, the manipulating Tor cells method is easy to deploy, accurate and effective. Similarly, in padding cell method, the time required to deanonymize hidden service is very high in comparison to other schemes. The required number of compromised nodes is also very low, which is a great benefit in terms of financial aspect. Cell counting based method gives real and significant threats to Tor's user, however, since it is completely dependent upon the amount of free memory in the system, it needs more time to make a buffer out of memory, especially for the system having large free amount of memory. Padding cell method is highly reliable but the probability to deanonymize hidden service by one of its guard node is 90% within 8 months [7].

Table 1: Comparison table of three different attacking schemes.

S.N.	Attacking Scheme	Simulation Environment	Deanonymizing Time	True Positive Rate	Required nos. of compromised entry nodes
1	Manipulating Tor cells method	Live Tor Network	low	100%	450 (30% of entire tor Entry router)
2	Cell counting based method	Shadow Simulator	medium	100%	10
3	Padding cell method	Live Tor Network	high	90%	1

IV. CONCLUSION

The hidden service inside Tor network is a double-edged sword. In one hand, it preserves the anonymity of the service provider, while on the other hand, it technically protects malicious users and organizations who host illegal contents such as drug and arms trading information, child pornography. A system, which can track down hidden service, is thus required, which will effectively deter malicious users from abusing the Tor network. In this paper, we outlined these schemes. From this, we were able to know how the attacks are used to deanonymize hidden services. From above comparison, we observed that, performance and the time are inverse to each other. If hidden service needs to be revealed in short time, then the required number of compromised entry nodes is higher which ultimately increases in the financial aspect.

ACKNOWLEDGEMENT

The work presented in this paper was supported by Research Funds of the Joint Research Program from the Attached Institute of ETRI in Korea. The authors express their appreciation for this support.

REFERENCES

[1] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The Second-Generation Onion Router," In proceedings of the 13th USENIX Security Symposium, pp. 303-320, August 2004.

[2] The Tor Project, Inc., "Tor: Anonymity Online," https://www.torproject.org/, 2012.

[3] "Silk Road (marketplace)," http://en.wikipedia.org/wiki/ Silk_Road_(marketplace), 2012.

[4] Z. Ling, J. Luo, W. Yu, X. Fu, D. Xuan, and W. Jia, "A new cell-counting-based attack against Tor," IEEEACM Trans. Netw. TON, vol. 20, no. 4, pp. 1245–1261, August 2012.

[5] Z. Ling, J. Luo, K. Wu, and X. Fu, "Protocol-level hidden server discovery," in INFOCOM, 2013 Proceedings IEEE, pp. 1043–1051,2013.

[6] R. Jansen, F. Tschorsch, A. Johnson, and B. Scheuermann, "The Sniper Attack: Anonymously Deanonymizing and Disabling the Tor Network," in Network and Distributed Systems Security Symposium (NDSS), San Diego, CA, USA, February 2014.

[7] A. Biryukov, I. Pustogarov, and R. Weinmann, "Trawling for tor hidden services: Detection, measurement, deanonymization," in Security and Privacy (SP), IEEE Symposium on, pp. 80–94, May 2013.

[8] "Configuring Hidden Services for Tor," https://www. torproject.org/docs/tor-hidden-service.html.en

[9] "Dutch police infiltrate hidden child porn websites in the U.S." http://lincolntribune.com/?p=19100, 2011.

[10] "TorDir - the link list /and pm system/ of tor," http://dppmfxaacucguzpc.onion/index.php?p=cat&cid=2 &sid=o40h5p0sllf4nlatcd0uo31377,2012.

[11] T. Dierks and C. Allen, "The TLS Protocol — Version 1.0," IETF RFC 2246, January 1999.

[12] "Tor Metrics Portal," https://metrics.torproject.org.

[13] "Tor Directory Protocol, Version 3," https://gitweb. torproject.org/torspec.git?a=blobplain;hb=HEAD;f=dir-spec.txt, 2012.

[14] R. Dingledine and R. Saal, "Tor and censorship: lessons learned," 2011.

[15] R. Dingledine and N. Mathewson, "Tor Path Specific ation," https://gitweb.torproject.org/torspec.git?a=blob_ plain ;hb=HEAD;f=path-spec.txt, 2008

[16] L. Øverlier and P. Syverson, "Locating Hidden Servers," in Proceedings of the IEEE Security and Privacy Symposium (S&P), pp. 100-114, May 2006.