A Novel Flow Multiplication Attack against Tor

Xiaogang Wang[1,2], Junzhou Luo[1], Ming Yang[1], Zhen Ling[1]

[1]*School of Computer Science and Engineering, Southeast University, Nanjing, P.R. China*
{wxiaog; jluo; yangming2002; zhen_ling}@seu.edu.cn
[2]*Changzhou College of Information Technology, Changzhou, P.R. China*

Abstract

Tor has become one of the most popular overlay networks for anonymizing TCP traffic. A novel and effective flow multiplication attack against Tor is proposed in this paper, which exploits the fundamental vulnerability of anonymous web browsing by using a man-in-the-middle attack on client's HTTP flow. In the flow multiplication attack, whenever a malicious exit onion router detects a web request to a target server, it responds with a malicious page embedded with specified number of image tags, which will cause the browser to initiate deterministic number of web connections on the same circuit to fetch those images. The entry onion router on the circuit can then find such traffic pattern and the communication relationship between the client and the web server will be discovered. Even if all active content systems such as JavaScript in the browser are disabled, our attack can still compromise the anonymity of Tor while achieving invisibility by keeping client's communication running continuously. The experiment results on Tor validate the feasibility and effectiveness of our attack.

Keywords: Flow Multiplication Attack, Tor, Anonymity, Traffic Pattern

1. Introduction

With the rapid growth and public acceptance of the Internet and the pervasive deployment of various wireless technologies, anonymity has become a necessary and legitimate aim in many applications including anonymous web browsing, instant messaging and location-based services (LBS). In these applications, encryption alone cannot maintain the anonymity required by participants. In order to provide anonymity to the communicating parties, extensive efforts have been made to design practical and effective anonymous communication systems.

Anonymous communication systems were first introduced by Chaum as early as 1981 [1]. However, it was until much later that a practical software system (Tor [2]) that enables its users to communicate anonymously on the Internet was introduced. Tor has become one of the most popular overlay networks for anonymizing TCP traffic. Tor employs low-latency anonymous scheme to support TCP applications such as web browsing on the Internet. The strength of the scheme is the use of strong, layered cryptography combined with a wide network of onion routers located across different administrative domains. By the end of 2008, there are over 2000 onion routers operating around the world, which form an overlay-based mix network. We use the terms router and onion router synonymously throughout this paper.

Recently extensive research work has been done on various attacks including traffic analysis attacks [3,4,5,6,7] and non-traffic analysis attacks to compromise the anonymity of Tor. Traffic analysis attacks are based on traffic analysis to determine whether Alice is communicating with Bob through Tor. Non-traffic analysis attacks [8] exploit the internal vulnerabilities of Tor's design.

While some countermeasures have been proposed to defend against such attacks, there is little research efforts that have been made to launch various application attacks against Tor. Specifically, web browsing [9] may initiate multiple web connections to fetch malicious invisible images embedded in a web page. These multiple web connections will form a specified traffic pattern and can be detected and used to launch a novel traffic analysis attack.

In this paper, we present a new attack against Tor, the flow multiplication attack, which is invisible and can confirm the communication relationship between the client and web server quickly and accurately, posing a serious threat against Tor. We assume that an attacker can control multiple routers, similar to other existing attacks [3,6]. When a client is accessing a target web server, a malicious exit router will responds with a malicious web page containing specified number of malicious image tags with invisible attributes and a meta refresh tag. Upon receiving such a malicious page, the web browser initiates deterministic number of web connections to fetch those corresponding images immediately. An accomplice of the attacker at the entry router on the circuit will detect such a designated traffic pattern, and the communication relationship between the client and the target web server is discovered.

We implement the flow multiplication attack on Tor and our experiments validate the feasibility and effectiveness of the attack.

The remainder of this paper is organized as follows: We review previous work in Section 2, introduce the

978-1-4244-3535-7/09/$25.00 ©2009 IEEE

basic process of web browsing through Tor in Section 3. We present the details of the flow multiplication attack in Section 4. In Section 5, we present our experiments on Tor to validate our findings. We conclude the paper in Section 6.

2. Related work

A number of low-latency anonymous communication systems [2,10,11] have been developed based on proxies or mixes. Notably, Tor [2] uses public key encryption on pre-determined sequence of onion routers to protect the transport of TCP flows, providing both sender anonymity and receiver anonymity. With the rapid growth and public acceptance of Tor, there have been more efforts to attack Tor. Such attacks consist of traffic analysis attacks and non-traffic analysis attacks.

Traffic analysis attacks are based on traffic analysis [3,4,5,6,7] to determine whether Alice is communicating with Bob through Tor. The traffic analysis attacks measure the similarity between the sender's outbound traffic and the receiver's inbound traffic in order to confirm their communication relationship. Levine et al. [4] utilized a cross-correlation technique for the traffic similarity measurement. Zhu et al. [5] proposed the scheme of using mutual information for the traffic similarity measurement.

Murdoch et al. [6] presented a low cost traffic analysis technique that allowed an outside observer to infer which routers are being used to relay a circuit's traffic, but could not trace the connection to the initiating client. Overlier et al. [3] studied an attack that uses one compromised router to generate false resource claims to attract traffic. Their approach is also based on the traffic timing similarity to associate circuits, and thereby locate the hidden servers in Tor network. Bauer et al. [12] studied an attack that is based on some malicious routers and used traffic analysis techniques. There are other traffic analysis attacks that intend to achieve both accuracy and invisibility [13].

Non-traffic analysis attacks exploit internal vulnerabilities of Tor's design itself. Murdoch [14] investigated an attack to reveal hidden servers of Tor by exploiting the fact that the clock deviations of a target server should be consistent with the server's load. The replay attack [8] against Tor exploits the fundamental design of Tor by employing duplicating cell approach to disrupt the normal counter at the middle and exit routers. Although such a non-traffic analysis attack can confirm the communication relationship quickly, it will break the anonymous communication circuit and degrade the performance of Tor.

Recently some attacks are proposed to attack web browsing by exploiting active content systems such as JavaScript, but they can be easily mitigated by disabling the active content systems in browser. Although some anonymity protection tools such as Privoxy software can be used to act as a web proxy with advanced filtering capabilities for protecting privacy, the features of Hyper Text Transport Protocol combined with compromised routers in Tor may cause new powerful threat to anonymity. The flow multiplication attack proposed in this paper is the attack exploiting the vulnerability of web browsing via Tor. This attack is powerful, capable of detecting one single web request accessing target web server. More attention needs to be focused on defending against such application attacks.

3. Web browsing through Tor

In this section, we introduce the Tor network, the process of constructing a circuit, and the process of web browsing through Tor.

3.1. The Tor network

The Tor network aims to provide end-user anonymity with constraints such as low-latency, deployability, usability, flexibility, and simple design. Currently, Tor can anonymize TCP streams, being a relatively high-throughput and low-latency onion routing network [2]. Figure 1 shows the fundamental architecture of the Tor network.

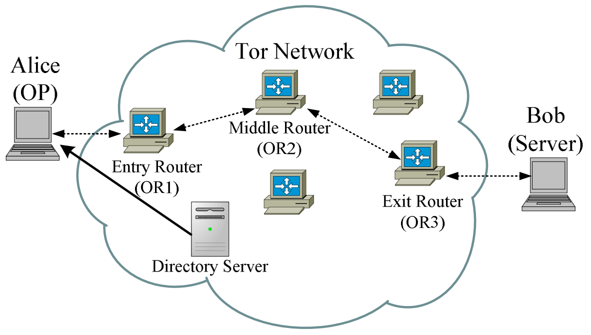

Figure 1. Tor network

Alice (i.e. Client) runs local software called an onion proxy (OP) to fetch router directories from directory servers, establish circuits across the network, and handle connections from user applications.

Bob (i.e. Server) runs TCP applications such as a web service and anonymously communicates with Alice.

Onion routers (OR) are special proxies that relay the application data between Alice and Bob. Transport Layer Security (TLS) connections are used for the overlay link encryption between two routers. The application data is packed into equal-sized cells carried through TLS connections. A circuit is a path of three routers (by default) through the Tor network from the proxy to the desired destination server. The first router on the circuit is referred to as the entry router, the second router is called a middle router, and the final hop is the exit router.

Directory servers include directory authorities and directory caches for maintaining and providing onion router information to clients. Functions of onion proxy, onion router and directory servers are all integrated into the same Tor software package. A user can edit a configuration file and configure a computer to have any combination of those functions.

Figure 2 illustrates the Tor cell format. All cells have a header including a circuit identifier and a command, which is not encrypted in the onion-like fashion so that the intermediate Tor routers can see this header. The other 509 bytes are encrypted in the onion-like way. There are two types of cells: control cell and relay cell. The control cell commands (*CMD*) includes *create or created* (used to set up a new circuit), *destroy* (to tear down a circuit) etc. The equal-sized relay cell has an additional header (the relay header) at the front of the payload, containing a stream identifier (*StreamID*), an end-to-end checksum for integrity checking (*Digest*), the length of the payload (*Len*), and a relay command (*CMD*). The entire contents of the relay header and relay cell payload are encrypted or decrypted together as the relay cell moves along the circuit. There are numerous types of relay commands that routinely traverse the circuit, such as *relay extend* (to extend a hop), *relay begin* (to open an application stream), *relay data* (to forward a stream), *relay end* (to close a steam cleanly) etc. We will explain them in later sections.

2	1	509
CircID	CMD	Data

2	1	2	6	2	1	498
CircID	Relay	StreamID	Digest	Len	CMD	Data

Figure 2. Tor cell format

3.2. Process of constructing a circuit

To anonymously access a web server, a client uses a way of source routing and chooses a series of routers from the locally cached directory, downloaded from the directory caches, to construct a circuit incrementally, negotiating a symmetric key with each OR on the circuit, one hop at a time.

1) Choosing three routers: As the default length of a circuit is three, the OP chooses three routers to begin constructing a new circuit. It first chooses an appropriate exit router which should have an exit policy supporting the relay of the TCP stream from the sender. Then, OP chooses an appropriate entry router and a middle router.

2) Constructing a three-hop circuit: First, OP issues a circuit building request to its chosen entry router and the entry router sends an acknowledgment. Next, OP sends another circuit building request to the entry router to extend the circuit to a chosen middle router. The middle router acknowledges the new circuit by sending an acknowledgment back to the client via the entry router. Finally, OP sends a request to extend the circuit to the chosen exit router, which is forwarded through the entry and middle routers to the chosen exit router. Once the exit router's acknowledgment has been received through the middle and entry routers, the circuit has been successfully built.

3.3. Web browsing over Tor circuit

We use web browsing shown in Figure 3 as the example to illustrate how a TCP stream is tunneled through the circuit that has already been created. When Alice wants to access a web server (i.e. Bob), Alice's application (i.e. web browser) asks her OP, which is implemented as a SOCKS proxy locally, to transmit web commands to Bob.

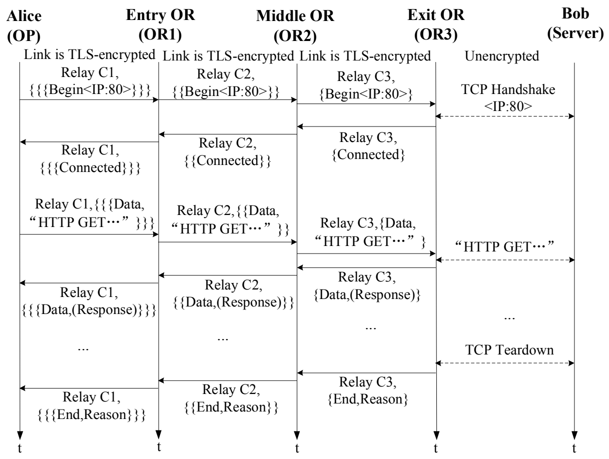

Figure 3. Alice accesses Bob's web server

Upon knowing the destination IP address and port from web browser, OP sends a *relay begin* cell to the exit router OR3, and the cell is encrypted as $\{\{\{Begin < IP, Port>\}_{k3}\}_{k2}\}_{k1}$, where the subscript refers to the key used for encryption of one onion skin. The three layers of onion skin are removed one by one each time the cell traverses a router through the circuit. When OR3 removes the last onion skin by decryption, it recognizes the request of opening a TCP stream to the port at the destination IP, which belongs to Bob. Therefore, OR3 acts as a proxy and builds a TCP connection with Bob. Once the OR3 connects to the web server, it sends a *relay connected* cell back to Alice's OP. The OP then accepts data from Alice's web browser and sends a *relay data* cell to the exit router OR3, and the cell is encrypted as $\{\{\{Data, \text{“HTTP, GET…”}\}_{k3}\}_{k2}\}_{k1}$.

When OR3 recognizes the *GET* request by decrypting the cell, it asks Bob to return the required web page, and sends a *relay data* cell back to Alice's OP as a response to OP's web request. The whole process is transparent to Alice, although she needs to configure her application to use the OP.

When Bob's web server finishes delivering data on a TCP connection to OR3, it will close the TCP connection. OR3 then sends a *relay end* cell along the circuit to OP and sends nothing more along the circuit for that stream. In the current Tor implementation, if the exit router receives an unrecognized cell, an error has occurred, and the circuit is torn down.

4. Flow multiplication attack

4.1. Attack process

As the goal of the flow multiplication attack is to confirm whether Alice is accessing Bob's web server, four important steps are required in the attack process.

1) Detecting target destination relay cell: The attack starts from the malicious exit router. When the exit router finds that OP is accessing target web server by checking the received *relay begin* cell, it first records the circuit identifier and stream identifier locally, then responds with a *relay connected* cell to OP. OP then sends the data from client's web browser in a *relay data* cell to the exit router. The cell is encrypted as $\{\{\{Data, \text{“HTTP, GET… ”}\}_{k3}\}_{k2}\}_{k1}$. Once OR3 detects such *relay data* cell from the marked circuit identifier and stream identifier, it logs the current time and responds with a malicious page shown in Figure 4.

```
<html>
<head>
<meta http-equiv="refresh" content="w;url= URLBob">
</head>
<body>
<img src=" URLBob/img1.gif" style="display:none"/>
<img src=" URLBob/img2.gif" style="display:none"/>
<img src=" URLBob/img3.gif" style="display:none"/>
... ...
<img src=" URLBob/imgm.gif" style="display:none"/>
</body>
</html>
```

Figure 4. Malicious web page

2) Initiating multiple web connections: Upon receiving such *relay data* cells containing the malicious page, the web browser immediately asks OP to initiate multiple web connections to fetch those required images during the specified time interval ***w***. As the body of the page contains nothing except multiple empty *img* tags with invisible attributes, only a blank browser window is prompted before Alice until the end of specified time interval ***w***. OP will actually initiate new multiple web connections by sending multiple *relay begin* cells to OR3 along the circuit. When these cells traverse the circuit and arrive at OR3, OR3 knows that such *relay begin* cells are only used to generate multiple streams. Therefore, OR3 responds with a *relay connected* cell for each *relay begin* cell. Similarly, upon receiving each *relay data* cell requiring corresponding image, OR3 will responds with an empty image file for each request and a *relay end* cell to close the connection. When the specified time interval ends, the web browser will initiate a new redirection connection towards its original web server (i.e. Bob's web server).

3) Detecting designated traffic pattern: At an entry router, an attacker needs to carefully detect the traffic flow consisting of forward and backward cells during the specified time interval. Although the entry router can't recognize the content of each cell traversed through it, it has the ability to check the circuit identifier and cell command by decrypting each cell, and can sample such obvious traffic pattern. The specified number of *img* tags is **m**, and each connection triggered by an *img* tag is completed with two relay cells in one direction, and another three in reverse direction. The designated traffic pattern consists of 5**m** relay cells traversed through the entry router and can be detected with the matched filter based approach to capture a segment of traffic cells for the best traffic match. Once the designated traffic pattern is detected, the entry router will log the source IP and the time of the occurrence of such traffic pattern. Such mechanism can exclude the circuit setup process, as it is unlikely that OP initiates connections matching the traffic pattern during the time interval. Such mechanism can also exclude normal anonymous web browsing process, as the probability of having 2**m** relay cells forwarded and **3m** relay cells transmitted backwards during the specified time interval for OP is very small. The time interval specified in the malicious page must ensure that all image connections specified in the malicious page can be completed.

4) Confirming the communication relationship: We use Network Time Protocol (NTP) to synchronize the time of the entry router and exit router. By correlating the time of injecting the malicious page with the time of detecting the designated traffic pattern, we can confirm that designated traffic pattern is actually caused by injecting malicious web page at the exit router. In addition, since the entry router knows the client of the TCP stream and the exit router knows the web server of the TCP stream, the communication relationship between the client and the web server is confirmed.

4.2. Threat effect

To identify the communication relationship of a TCP stream, the attack is only based on one invisible malicious page. The detecting of target web request is quite simple and can be carried out quickly and accurately by the exit router. There is no need for our attack to select the cells of TCP stream data as the Replay Attack [8]. Notably, our attack is stealthy, as Alice can see nothing strange on her web browser except waiting for some additional time. The circuit built is still available. The performance of Tor will not be degraded more by our attack. This may prevent Tor directory authorities from monitoring the activities of routers and blacklisting those routers with misbehaved activities. So, our attack is more effective and poses great threat to the anonymity service provided by Tor even if all active content systems in the browser are disabled. Our attack can be used to randomly profile both clients and servers hidden by Tor.

4.3. Attack effectiveness

The success of our attack depends on the assumption that the attacker controls both entry and exit routers on the circuit. This is a reasonable assumption due to the principle of voluntary-oriented operation and Tor's router selecting scheme which prefers high-bandwidth, high-uptime routers in Tor. The attacker can compromise an unfair percentage of entry and exit routers by configuring its routers and exaggerating its resource claims. As long as a router has a self-designed exit policy enabling access to external services, this router becomes an exit router. If a router has a mean time between failures (MTBF) not less than the median for active routers or at least 10 days, it becomes a stable router. A stable router can be promoted to an entry guard if its bandwidth is either at least the median among known active routers or at least 250KB/s. This set of criteria is not difficult to meet in the real-world.

Let b_i be the bandwidth advertised by the i-th router, and assume that there are N routers, among which the first K routers are malicious. Then the probability p_i that the i-th router is chosen by OP is $b_i / (\sum_{j=1}^{N} b_j)$.

As a router can't be chosen twice for the same circuit, the probability S_{attack} for the attacker who controls both entry router and exit router successfully on a given circuit can be computed by the following formula.

$$S_{attack} = \sum_{i=1}^{K} p_i \cdot (\sum_{\substack{j=1 \\ j \neq i}}^{K} \frac{p_j}{1-p_i})$$

It is obvious that larger p_i (i=1,…, K) will lead to larger probability of S_{attack}. As long as p_i (i=1,…, K) gets large by exaggerating resource claims, the attacker can succeed to launch the flow multiplication attack with higher probability.

The detection rate of our attack may be influenced by Internet traffic dynamics and other relating traffic flows. For example, while the entry router is detecting the designated traffic pattern, it receives other unrelated flows during the specified time interval. It is also possible for some web page to initiate those flows exactly matching our designated traffic pattern during the time interval. Based on accessing 100 different web servers via Tor, we have not recorded such occasions. This confirms that the false positive rate of our attack is low. We also use a threshold-based traffic detection method to improve the effectiveness of our attack. Furthermore, we may choose appropriate number of images $\boldsymbol{m}$ and the time interval $\boldsymbol{w}$ to achieve high detection rate and low false positive rate.

5. Evaluation

We implement the flow multiplication attack based on the Tor release version of 0.2.0.31 to investigate the effectiveness and feasibility of the attack.

5.1. Experiment setup

We conduct experiments in a partially controlled environment as shown in Figure 5. The OP code is modified for debugging purposes to build circuits through the designated malicious exit router and entry router. The Tor client uses Internet Explorer 7.0 to access the target web site http://www.seu.edu.cn through the patched OP using Privoxy. By using the Tor configuration file and manipulating parameters, such as EntryNodes, ExitNodes, StrictEntryNodes, and StrictExitNodes, we setup the client to select the malicious routers along the circuit. The middle router is selected using the normal Tor routing selection algorithm. The entry router is configured to log the information of every cell traversed through it and revise its code to detect designated traffic pattern. The entry router's exit policy only allows it to be used as an entry or middle router. The exit router is designed to inject a malicious web page, and use the default exit policy from Tor.

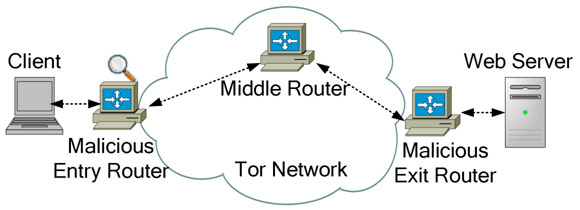

Figure 5. Experiment setup

5.2. Experimental results and Analysis

The client accesses the same web server every 20 seconds. The malicious web page contains 8 malicious empty *img* tags and the time interval is 10 seconds. The revised code for the exit router is implemented to inject a malicious web page whenever it detects a web request to the target server and records the time of injecting the malicious page. The revised code for the entry router records the time of detecting the designated traffic pattern and carries out the detection. We use the correlation coefficient ρ as proposed in [8] to measure the strength of correlation between the time of injecting the malicious page and the time of detecting the designated traffic pattern.

Correlation coefficient is defined in the following formula:

$$\rho_{x,y} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2} \cdot \sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

where x is the time of injecting the malicious web page at the exit router and y is the time of detecting the designated traffic pattern at the entry router. $\bar{x}$ and $\bar{y}$ are the mean values of x and y, respectively.

Figure 6 shows the relationship between the time of injecting the malicious page and the time of detecting the designated traffic pattern. It can be concluded from the figure that the actual correlation coefficient between them is close to one. This strongly confirms the effectiveness of our attack against Tor.

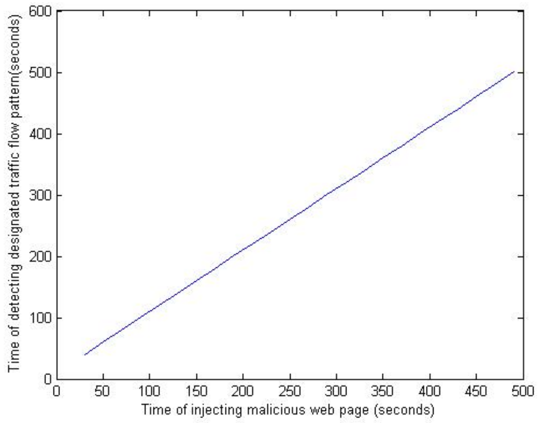

Figure 6. Correlation between time of injecting web page and time of detecting traffic pattern

6. Conclusion and future work

A novel flow multiplication attack against Tor proposed in this paper can achieve invisibility and accurate confirmation of the communication relationship by exploiting the flow multiplication features of web browsing, significantly degrading the anonymity service provided by Tor. Via real-world implementation and experiments, the effectiveness and feasibility of the flow multiplication attack are validated. Our study is critical for securing and improving Tor.

Due to the Tor's fundamental design, defending against the flow multiplication attack still remains a challenging task and we will investigate it in our future research. Also, we will investigate new attacks against Tor by exploiting other application features and develop corresponding countermeasures.

Acknowledgement

This work is supported by China Specialized Research Fund for the Doctoral Program of Higher Education under Grant No. 200802860031, Jiangsu Provincial Natural Science Foundation of China under Grant No. BK2007708 and BK2008030, Jiangsu Provincial Key Laboratory of Network and Information Security under Grant No. BM2003201, Key Laboratory of Computer Network and Information Integration of Ministry of Education of China under Grant No. 93K-9, and International Science and Technology Cooperation Program of China.

References

[1] D. L. Chaum, "Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms", *Communications of the ACM*, 1981, 24(2), pp. 422–426.

[2] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The Second-Generation Onion Router", *Proceedings of the 13th USENIX Security Symposium*, San Diego, CA, USA, Aug. 9–13, 2004, pp. 21-38.

[3] L. Overlier and P. Syverson, "Locating Hidden Servers," *Proceedings of the IEEE Security and Privacy Symposium (S&P),* Berkeley, California, USA, May 21-24, 2006, pp. 100-114.

[4] B. N. Levine, M. K. Reiter, C. Wang, and M. Wright, "Timing Attacks in Low-Latency Mix Systems," *Proceedings of Financial Cryptography (FC)*, Key West, Florida, USA, Feb. 9-12, 2004, pp. 251-265.

[5] Y. Zhu, X. Fu, B. Graham, R. Bettati, and W. Zhao, "On Flow Correlation Attacks and Countermeasures in Mix Networks," *Proceedings of Workshop on Privacy Enhancing Technologies (PET)*, Toronto, Canada, May 26-28, 2004. pp. 207-225.

[6] S. J. Murdoch and G. Danezis, "Low-Cost Traffic Analysis of Tor," *Proceedings of the IEEE Security and Privacy Symposium (S&P),* Oakland, CA, USA, May 8-11, 2005, pp. 183-195.

[7] X. Fu, Y. Zhu, B. Graham, R. Bettati, and W. Zhao, "On Flow Marking Attacks in Wireless Anonymous Communication Networks," *Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS),* Columbus, OH, USA, June 6-10, 2005, pp. 493–503.

[8] R. Pries, W. Yu, X. Fu and W. Zhao, "A New Replay Attack Against Anonymous Communication Networks", *Proceedings of the IEEE International Conference on Communications (ICC) - Information and Network Security Symposium*, Beijing, China, May 19-23, 2008, pp. 1578-1582.

[9] Q. Sun, D. R. Simon, Y. Wang, W. Russell, V. N. Padmanabhan, and L. Qiu, "Statistical Identification of Encrypted Web Browsing Traffic," *Proceedings of IEEE Symposium on Security and Privacy (S&P)*, Berkeley, California, USA, May 12-15, 2002, pp. 19-30.

[10] M. K. Reiter and A. D. Rubin, "Crowds: Anonymity for Web Transactions", *ACM Transactions on Information and System Security*, 1998, 1(1), pp. 66–92.

[11] M. J. Freedman and R. Morris, "Tarzan: A Peer-to-Peer Anonymizing Network Layer", *Proceedings of the 9th ACM CCS*, Washington, USA, Nov. 18-22, 2002, pp. 193-206.

[12] K. Bauer, D. McCoy, D. Grunwald, T. Kohno, and D. Sicker, "Low-Resource Routing Attacks Against Tor", *Proceedings of ACM Workshop on Privacy in the Electronic Society (WPES)*, Alexandria, VA, USA, Oct. 29-29, 2007, pp. 11-20.

[13] W. Yu, X. Fu, S. Graham, D. Xuan, and W. Zhao, "DSSS-Based Flow Marking Technique for Invisible Traceback", *Proceedings of the 2007 IEEE Symposium on Security and Privacy (S&P)*, Oakland, California, USA, May 20-23, 2007, pp. 18-32.

[14] S. J. Murdoch, "Hot or Not: Revealing Hidden Services by their Clock Skew", *Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS)*, Alexandria, Virginia, USA, Oct. 30-Nov. 3, 2006, pp. 27-36.