

# On Performance Bottleneck of Anonymous Communication Networks

Ryan Pries, Wei Yu, Steve Graham, and Xinwen Fu

## Abstract

Although a significant amount of effort has been directed at discovering attacks against anonymity communication networks and developing countermeasures to those attacks, there is little systematic analysis of the Quality of Service (QoS) for such privacy preserving systems. In this paper, we initiate the effort to study the QoS of Tor, a popular anonymous communication network on the Internet. We find that Tor suffers severe TCP performance degradation because of its random path selection strategy. Our investigation shows that Tor's bandwidth weighted path selection algorithm can only improve the performance to a very limited extent. We analyze this performance issue from the perspective of overlay networks and model the TCP throughput of Tor. We conduct extensive experiments on the real-world Tor network and the experimental results validate our theory. We also discuss possible remedies to this performance issue.

## Index Terms

Mix network, Anonymity, Overlay, TCP, QoS

## I. INTRODUCTION

Society has been moving toward a tetherless world over the past decade, as evidenced by the pervasive deployment of wireless technologies connecting to the Internet. While such technology provides remarkable convenience, the network infrastructure supporting it has become a target for criminals. Criminal attacks may range from disrupting services to theft of personal or business-critical data. One technological response to these threats is the development of privacy preserving techniques such as *mix networks* [1]. Privacy preserving techniques have been playing, and will continue to play, an important role in protecting users' online activities.

For security and privacy technologies to be useful in practice, there are various considerations that need to be addressed. We consider four key requirements: *usability*, *deployability*, *effectiveness*, and *robustness* [2]. Usability requires sufficient QoS in a technology so that desired online activities have acceptable performance. Deployability requires that costs in human and material resources required to place a technology in service is lower than the value such a technology provides. Effectiveness for privacy protection technologies is the capability of detecting or concealing users' connectivity as needed. Robustness requires the ability to survive, to some extent, unplanned challenges due to the environment, mis-configuration or operator error. Systems not meeting these requirements may not be suitable for practical deployment as a large-scale distributed system.

Researchers have been developing anonymous communication systems for the last two decades. Mix networks [1] are a popular approach to provide anonymity. Mix networks were developed for

Xinwen Fu (corresponding author), Steve Graham, and Ryan Pries are with the College of Business and Information Systems, Dakota State University, 820 N. Washington Ave. Madison, SD 57042 (Email:{xinwen.fu, skg, priesr}@dsu.edu). Wei Yu is with the Department of Computer Science, Texas A&M University, 301 Harvey R. Bright Bldg, College Station, TX 77843 (Email: weiyu@cs.tamu.edu).

This project was partially supported by NSF grants CNS-0721766 and CNS-0722856. Any opinions, findings, conclusions, and/or recommendations expressed in this material, either expressed or implied, are those of the authors and do not necessarily reflect the views of the sponsors listed above.

anonymous email [3] and have later been adopted to provide anonymity for low latency connection based applications such as web browsing [4]. We focus on low latency applications, where the demands for practical utility are more stringent.

Significant and widespread effort has been directed at discovering attacks against anonymous communication networks, as well as developing countermeasures for such attacks [3], [5], [6], [7], [8], [9], [10], [11], [12]. However, there is little systematic analysis of QoS for such privacy preserving systems, leaving the question of their practical utility largely open. Work on this topic [13] by McCoy *et al.* plainly presented results of some Tor [4] performance measurements, such as Tor router geopolitical distributions, circuit latency, circuit throughput (in the default Tor configuration), and traffic protocol distribution. Because of increasing demand and limited research, more extensive performance studies of privacy preserving systems are essential, and must address both theoretical and experimental performance. Additionally, the performance measures considered must include the effectiveness of such systems at preserving privacy in conjunction with other system performance metrics.

In this paper, we address this need by examining both the theoretical and empirical performance of Tor, a popular real-world anonymous communication network, primarily supporting TCP applications such as anonymous web browsing [4]. At the time that this paper was written, there are 1044 Tor routers running around the world, and those routers form an overlay mix network. In practice, Tor suffers severe TCP throughput degradation. Tor is currently not recommended for bulk data transmission (such as *FTP* or *Bittorrent*) by its developers [14]. In this paper, we identify and study the key factors causing such QoS degradation: Tor's random path selection strategy. Although the developers of Tor use a bandwidth weighted algorithm intending to improve the overall system performance, we find that the improvements due to the algorithm are very limited in practice because of the high percentage of low-bandwidth routers on the Tor network. Based on the overlay network model, we analyze path TCP throughput bounds for Tor. We conduct extensive experiments on the real-world Tor network which validate our theoretical analysis. In addition, we consider possible remedies for this performance issue.

The remainder of this paper is organized as follows: We present a model of TCP throughput in the Tor network and identify its performance bottleneck in Section II. We analyze the QoS of Tor and discuss possible solutions to this performance issue in Section III. Experimental results on Tor are presented in Section IV. We review related work in Section V and conclude this paper in Section VI.

## II. MODELS

In this section, we first introduce Tor from the perspective of overlay networks and then present a model of TCP throughput for the Tor network.

### A. Tor Overlay Network

Tor is an overlay network on the Internet providing anonymous communication. Based on its current design, Tor supports only TCP applications. Figure 1 shows the basic structure of the Tor network. Generally speaking, there are four entities within the Tor network as follows:

- *Client*: the user of the Tor network for private communication through the Internet. A client uses an *onion proxy (OP)* installed on her local machine and connects to services through the Tor network. *Client* may refer to either the user of a client system or the client system, as appropriate by context.
- *Server*: the target TCP applications such as web servers accessed by a client.
- *Tor router*: the special proxy which relays the application data for the client to the server. TLS connections are built between Tor routers for link encryption. The application data are

packed into equal-sized cells (512 bytes), which are carried as TLS records. In this paper, the terms *Tor router*, *onion router (OR)* and *routers* may be used interchangeably. The meaning of a term should be clear within the context.

- *Directory servers*: servers holding information regarding the onion routers. There are authority directory servers and their mirrors. The client downloads the router directory from directory servers and caches it locally.

Functions of onion proxy, onion router and directory server are integrated into the same software package. A user can edit a Tor's configuration file and configure a computer to have any combination of those functions. A volunteer donating her network bandwidth installs the package and configures her computer as an onion router or directory server, which uses different well known ports for protocol information exchange and data relay. Onion routers can be configured using a leaky bucket mechanism to constrain the donated bandwidth. A client installs the same software package but configures her computer as a Tor client to utilize the Tor network.

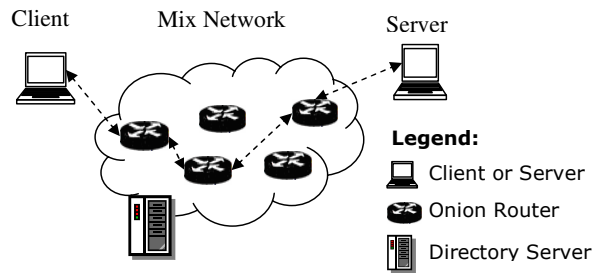


Fig. 1. Tor Network

The Tor network is a straightforward mix network. The basic operation is illustrated in Figure 1. To browse the web server in Figure 1 while hiding the connection, the client chooses a series of Tor routers from the cached Tor router directory. We refer to the sequence of ordered Tor routers as the *path* of the client's communication through the Tor network. The number of Tor routers is the *path length*. We will use a path length of three to describe Tor's operation. The client first negotiates session keys with the three chosen routers, one by one, using the *Diffie-Hellman* handshake protocol. The last Tor router is the *exit node*. The client packs application data into cells.

The cell is encrypted in a layered, onion-like way: the data is first encrypted with the third (exit node) session key shared with the 3<sup>rd</sup> Tor router, then with the 2<sup>nd</sup> session key and finally with the 1<sup>st</sup> session key (the outermost layer). When an encrypted cell passes through the Tor network, the first Tor router on the path peels off the outermost layer, which is encrypted using its session key for that path, and forwards the thinner, still onion-like cell to the next Tor router on the path. This process proceeds hop by hop along the circuit chosen by the client until the cell reaches the last Tor router, which acquires the necessary information about the requested service and acts as a proxy for communication with the server. Each Tor router checks the *exit flag* in the cell to determine if it is the exit node. The flag can only be detected at the pre-selected exit node, where the packet has been fully decrypted<sup>1</sup>.

The replies from the server are sent backward to the client in cells, which are encrypted hop by hop by the same Tor routers along the same circuit in the reverse order. The client will then decrypt those cells using the negotiated session keys. Note that except that the connection from the exit node to the server is not link encrypted, all other connections along the circuit are protected by TLS (Transport Layer Security), which carries Tor cells in TLS records.

<sup>1</sup>In this way, the encrypted cell has a fixed length and does not swell as in the public key encryption case [1].

### B. TCP Throughput in Overlay Networks

There are a few approaches to approximating the end-to-end TCP throughput [15], [16]. We use the following result from [15] for our estimations of end-to-end TCP throughput:

$$T \leq \frac{1.5\sqrt{1/3}B}{d\sqrt{p}}. \quad (1)$$

In Formula (1),  $T$  is the throughput,  $B$  is the packet size,  $d$  refers to the round trip time (TTL) including queuing delay, and  $p$  is the packet drop rate.

However, in a Tor network, the end-to-end (from the client to the server) TCP throughput cannot be directly estimated by Formula (1). Since Tor is an overlay network, a set of sequential TCP connections are used to relay packets from the source to the destination. Figure 2 shows a path of length 3. If the client downloads a file from the server, a packet is relayed by four separate TCP flows along the four segments:  $L_1 = \langle v_0, v_1 \rangle, \dots, L_4 = \langle v_3, v_4 \rangle$  with packet drop rate and round trip delay pairs  $\langle p_1, d_1 \rangle, \dots, \langle p_4, d_4 \rangle$ , for each segment. If any of these segments becomes a bottleneck, the end-to-end throughput will be limited by the bottleneck segment [17].

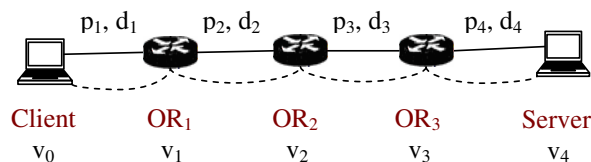


Fig. 2. TCP Throughput Along Path in the Tor Network

Let  $\Omega$  be the set of Tor nodes in the Tor network. A path of length  $m$  will select  $m$  Tor nodes,  $V = \{v_1, \dots, v_m\}$  from the Tor node set  $\Omega$ , where  $v_i \in \Omega$ .  $v_0$  will be the client and  $v_{m+1}$  will be the server, so the whole path is  $V = \{v_0, v_1, \dots, v_m, v_{m+1}\}$ .  $T_i$  is the throughput along the segment from  $v_{i-1}$  to  $v_i$ , so the TCP throughput along the path,  $T_{path}$ , is the minimum of  $T_i$  ( $i \in [1, m]$ ) as follows:

$$T_{path} = \min_{i=1}^{m+1} T_i. \quad (2)$$

In summary, Tor is an overlay network and it relays the traffic from one Tor router to another. The TCP throughput along the path on Tor is approximated by Formula (2).

### III. ANALYSIS OF TCP PERFORMANCE IN TOR

In this section, we demonstrate theoretically that the poor QoS of the Tor network comes from the algorithm for randomly choosing routers along a path. We first examine the impact of the random router selection on Tor's TCP throughput and then introduce the bandwidth weighted path selection algorithms which may be used by the Tor network. Finally, we discuss some possible remedies to improve Tor's QoS.

#### A. Impact of Random Router Selection on TCP Throughput

Two important factors affecting the anonymity provided by a mix network are the path selection algorithm and the path length. Researchers have not reached consensus on the algorithms for determining paths and path length for optimizing the trade-off between anonymity and performance for anonymous communication networks in practical use, such as Tor. The intuitive path selection algorithm creates a path by randomly selecting routers. This approach is used for part of the path creation process by Tor. The default path length used by many anonymous communication

networks including Tor is three [4], [18]. In the following, we analyze the impact of path length on TCP Throughput when using the random path selection algorithm with the configurable path length [19].

Recall that Tor routers use donated bandwidth from users, who may limit the donated bandwidth using the leaky bucket mechanism. Hence, it is reasonable to assume Tor routers in Figure 1 are bottlenecks for TCP throughput and that the client and server nodes are not. Our experiments confirmed this assumption. Lemma 1 gives a bound for average TCP throughput along the path. This bound refers to the case where the sequential TCP connections of a Tor path have exclusive access to the Tor bandwidth. The bandwidth available for a TCP connection may be much lower than this bound because the links along the path may be shared by cross traffic. We derive the theoretical bound in order to demonstrate the impact of the path length on Tor's TCP throughput. Empirical data from our experiments described in Section IV corroborate the theory.

*Lemma 1:* In a Tor network, there are  $n$  Tor Routers, and the path length is  $m$ . Assume that the bandwidths of the routers comprise a set  $\{B_1, \dots, B_n\}$  and  $B_1 > \dots > B_n$ . So  $B_n$  is the minimum bandwidth in the set. The bound for average TCP throughput along a path in the Tor network can be calculated by

$$B(n, m) = \frac{\sum_{k=m}^n \binom{k-1}{m-1} B_k}{\binom{n}{m}}. \quad (3)$$

**Proof:**

As shown in Formula (2), within an overlay network such as Tor, the end-to-end TCP throughput along the path is determined by the minimum throughput of any segment of the path. Thus if  $B_n$  is chosen as part of a path,  $B_n$  is the throughput bound along that path. Let's calculate the number of length  $m$  paths including  $B_n$ . Since the path length is  $m$  and  $B_n$  is chosen, we need to choose  $m - 1$  nodes from among the remaining  $n - 1$  nodes. There are  $\binom{n-1}{m-1}$  choices. Hence, the probability of path throughput being  $B_n$  is  $\binom{n-1}{m-1} / \binom{n}{m}$ . If a path throughput bound is  $B_{n-1}$ , then  $B_n$  cannot have been chosen for that path. There are  $\binom{n-2}{m-1}$  possible paths of length  $m$  with bandwidth  $B_{n-1}$ . Thus the probability of path throughput being  $B_{n-1}$  is  $\binom{n-2}{m-1} / \binom{n}{m}$ . For the general case, if  $B_k$  is the path throughput bound, there are  $\binom{k-1}{m-1}$  possible paths of length  $m$  with throughput bound  $B_k$ . Then the probability of choosing  $B_k$  is  $\binom{k-1}{m-1} / \binom{n}{m}$ . Therefore, the bound for the average TCP throughput along a path in the Tor network can be derived by

$$\begin{aligned} B(n, m) &= \frac{\binom{n-1}{m-1}}{\binom{n}{m}} B_n + \frac{\binom{n-2}{m-1}}{\binom{n}{m}} B_{n-1} + \dots + \frac{\binom{m-1}{m-1}}{\binom{n}{m}} B_m \\ &= \frac{\sum_{k=m}^n \binom{k-1}{m-1} B_k}{\binom{n}{m}}. \end{aligned} \quad (4)$$

■

In Lemma 1, we assumed that  $B_1 > \dots > B_n$ . In practice, bandwidths are rarely the same and the assumption holds. From the proof, it can be observed that the argument still holds if  $B_1 \geq \dots \geq B_n$ , and Lemma 1 remains true. From the proof of Lemma 1, we can see that as the path length increases, the probability that a Tor router with low bandwidth is chosen increases. Thus longer path lengths will reduce the overall TCP throughput of the Tor network. Theorem 1 formalizes this statement.

*Theorem 1:* When the path length increases, the bound for the average TCP throughput along a path in the Tor network decreases. That is,

$$B(n, m - 1) \geq B(n, m). \quad (5)$$

**Theorem 1 states that a longer path reduces the path TCP throughput, because choosing a greater number of routers increases the probability that a low-bandwidth Tor router is chosen.** The detailed proof of Theorem 1 can be found in Appendix A.

### B. Tor's Bandwidth Weighted Routing

The developers of Tor have been developing various path selection algorithms and adjusting path lengths in order to improve the trade-off between Tor's QoS and anonymity. Researchers have not concluded that any choice is definitely better than the other possibilities. Specifically, Algorithms 1, 2, 3 and 4 describe the basic default path selection algorithms in the implementation of Tor (release version 0.1.1.26). We determined these algorithms by reviewing Tor's source code [19]. The algorithms can take many user preferences and options into consideration, but primarily choose nodes randomly or use a bandwidth weighted pseudo-random method [19], [20]. The concept behind bandwidth weighted selection is to preferentially use Tor nodes with higher bandwidth, and as a result achieve better QoS<sup>2</sup>.

Algorithm 1 presents the path selection algorithm which chooses routers in a specific order given a desired path length of  $m$ . A path of length  $m$  will select  $m$  Tor nodes,  $V = \{v_1, \dots, v_m\}$  from set  $\Omega$ , where  $\Omega$  is used to denote the set of all Tor nodes. Tor does not choose the same router twice for the same path [19]. Algorithm 2 presents the selection of the exit node for a circuit. A bandwidth weighted algorithm is used. Algorithm 3 describes the selection of the entry node for a circuit. Note that if a chosen entry node fails later, a new entry node will be chosen [19]. Algorithm 4 describes the selection of middle nodes for a circuit. The middle nodes can be chosen based upon a bandwidth weighted algorithm. The weighted algorithm [19] used by Tor is ad-hoc. But it does not influence the overall observations we have made as a result of the theory in Section III-A.

From these algorithms, we know that the bandwidth weighted algorithm has been extensively applied to select both the exit node and middle nodes. However, our experiments on Tor in Section IV still show very poor TCP performance due to the factors discussed in Section III-A. In Section IV, we show that Theorem 1 still matches the experimental results on the Tor network well. This is a result of the distribution of bandwidths among the routers on the Tor network. In particular, a significant percentage of the routers in daily use have limited bandwidth. For example, from Figure 4 in Section IV, we know that 50% of the Tor routers have bandwidth lower than  $50KB/s$ . The probability of choosing one of these low-bandwidth routers will be still high. Consider the following simple example. Assume a set of four routers with bandwidths of  $100KB/s$ ,  $50KB/s$ ,  $50KB/s$ , and  $50KB/s$ . If we choose the middle router along a path, a router with  $50KB/s$  is chosen with a probability of 60% based on the bandwidth weighted algorithm.

---

#### Algorithm 1: Tor path selection

---

- 1 The exit node is chosen as node  $v_n$  based on Algorithm 2;
  - 2 The entry node (or entry guard) is chosen as node  $v_1$  based on Algorithm 3;
  - 3 The remaining  $m - 2$  middle nodes are chosen as nodes  $v_2, \dots, v_{n-1}$  based on Algorithm 4.
- 

<sup>2</sup>This may lead to routing-based attacks on anonymity [13].

---

**Algorithm 2:** Selection of the exit node for a circuit
 

---

- 1 Exclude the client (in case that client is an OR node) as an possible exit node;
  - 2 Exclude non-running nodes and router not meeting capacity and uptime requirements;
  - 3 Exclude routers whose policy rejects all;
  - 4 Derive a list of qualified exit nodes with bandwidth  $\{B_{e,1}, \dots, B_{e,m}\}$ ;
  - 5 Use the bandwidth weighted algorithm to choose the exit node such that the node with bandwidth  $B_{e,j}$  is chosen with the probability  $B_{e,j} / \sum_{i=1}^m B_{e,i}$ .
- 

---

**Algorithm 3:** Selection of the entry node for a circuit
 

---

- 1 Derive a list of entry nodes;
  - 2 Select a default number 3 of them as entry nodes for the client.
- 

### C. Discussion

From our analysis and experimental measurements in Section IV, we know that the bandwidth weighted algorithm used by Tor provides limited improvement of its TCP throughput. In this section, we discuss the alternatives which might improve performance on the Tor network.

From Lemma 3, it is easy to see that if the distribution of donated bandwidth is shifted toward higher bandwidths, overall Tor performance will increase. This suggests that a dedicated anonymous communication backbone of higher performance routers may dramatically improve future anonymous communications. Similarly, if we use fewer low-bandwidth Tor routers, we can improve overall TCP throughput on the Tor network. The Tor network could be partitioned into classes of Tor routers with high or low donated bandwidth. Paths drawn from the class of high-bandwidth routers can provide markedly better performance, while those from the class of low-bandwidth routers have throughput no lower than unpartitioned paths, lower throughputs are more probable. This fact is derived in Theorem 2. The detailed proof can be found in Appendix B.

*Theorem 2:* In a Tor network, there are  $n$  Tor routers, and the path length is  $m$ . Assume that the bandwidth of Tor routers forms a set  $\{B_1, \dots, B_l, B_{l+1}, \dots, B_n\}$  and  $B_1 > \dots > B_l > B_{l+1} > \dots > B_n$ . If only the first  $l$  Tor routers are used, the average of the path TCP throughput increases. That is,

$$B(l, m) > B(n, m), \text{ where } l < n. \quad (6)$$

Nonetheless, there are a few concerns with using fewer low-bandwidth Tor routers. First, Tor routers with high bandwidth may be congested, degrading overall TCP throughput again. Second, the use of fewer routers may reduce the anonymity that Tor provides. While the first point is obvious, we can develop schemes to select low bandwidth Tor servers with markedly lower probability as a means of addressing congestion and of preserving anonymity. In the mean time, router path selection can be improved by adopting additional constraints. For example, when selecting a middle Tor router, the bandwidth of entry and exit nodes along the same path should be considered. If one such node is of low bandwidth, selecting a middle Tor router with high bandwidth will waste resources based on Formula (2). The second point is arguable. When more traffic from different users are mixed together through a smaller number of Tor routers, it actually increases the difficulty for attackers to carry out traffic analysis attacks [21]. Hence, a congested Tor network may achieve better anonymity despite degraded performance. This shows the dilemma of trade-offs between anonymity and QoS.

In addition, to better use the available bandwidth contributed by Tor routers, it is desirable to let the Tor routers update the run-time bandwidth information, which is widely used in QoS routing

---

**Algorithm 4:** Selection of middle nodes for a circuit
 

---

- 1 Exclude the exit node and entry node on the circuit;
  - 2 Exclude the client (in case that client is an OR node) as the possible middle node;
  - 3 Derive a list of qualified running nodes;
  - 4 **if** *Fast circuits required* **then**
  - 5 |   Use a bandwidth weighted algorithm to choose middle nodes;
  - 6 **else**
  - 7 |   Choose middle nodes randomly.
  - 8 **end**
- 

schemes [22]. In this way, the path selection algorithms can choose better candidates reflecting real bandwidth utilization. Partitioning routers based on bandwidth can also pave the way to improved performance by supporting differential QoS in the Tor network. In particular, the routers could be organized into multiple classes based on their bandwidth and chosen for flow requests based on a particular flow request’s priority. In this way, higher priority flows will obtain high bandwidth and low priority flows will obtain lower bandwidth. So long as user requirements can be met with differential QoS, this will make more effective use of bandwidth.

#### IV. EVALUATION

In this section, we report the results of experiments using the real-world Tor network which validate the theoretical results in Section III.

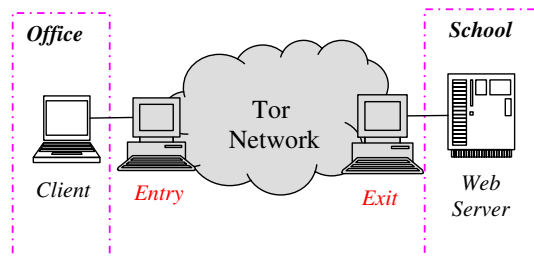


Fig. 3. Experiment Setup

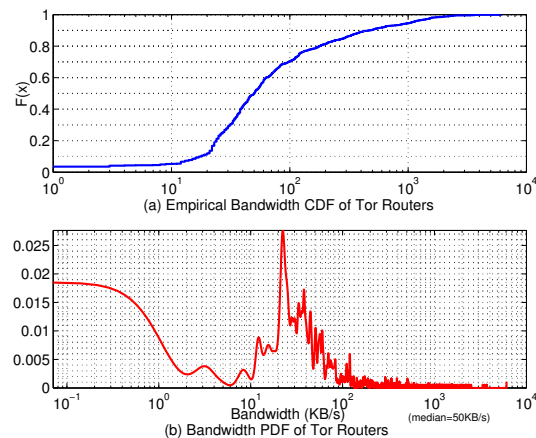


Fig. 4. Bandwidth CDF and PDF of Onion Routers

##### A. Experiment Setup

Figure 3 gives the experiment setup<sup>3</sup>. The client in a school office downloads a file of size 458,752 bytes from the same school’s web server using a path through the Tor network. The downloading tool was the command line utility *wget*, shipped with most Linux distributions. By configuring *wget*’s *http\_proxy* and *ftp\_proxy* parameters, we can force *wget* to download files through *Privoxy*, the proxy server used by *Tor* at the client site.

<sup>3</sup>The school in Figure 3 refers to Dakota State University.

We revised Tor release 0.1.1.26 in order to control the use of specified nodes as entry, middle, and exit routers as needed for our experiments. Tor provides options of *EntryNodes* and *StrictEntryNodes* to specify the entry nodes, and *ExitNodes* and *StrictExitNodes* to specify the exit nodes<sup>4</sup>. We modified the use of the *TestVia* option of Tor to allow us to specify the middle Tor routers. With these options, all Tor circuits built from the client, can be controlled, permitting repeatable empirical experiments.

### B. Experimental Results

To quantify the impact of a slow router on Tor throughput, we conducted a series of experiments. Figure 4 shows the bandwidth distribution of Tor routers based on the information retrieved from Tor directory authorities on September 9, 2007. We can see that the median is only  $50KB/s$  and 70% of the bandwidth is lower than  $100KB/s$ .

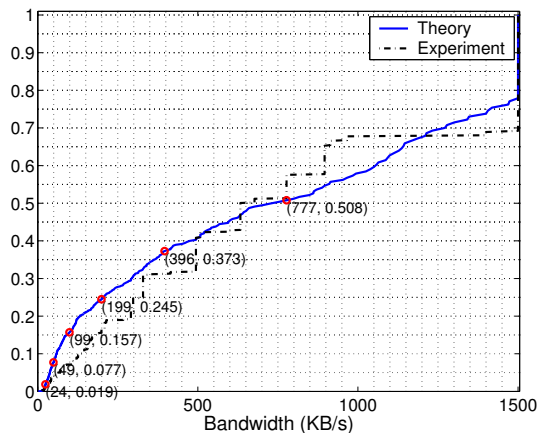


Fig. 5. Summary of Tor's Bandwidth Distribution

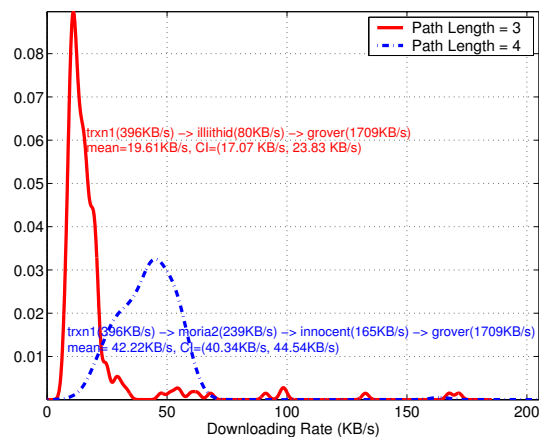


Fig. 6. Slow Link's Impact on TCP Throughput (the number in the round bracket following a router name is its observed bandwidth)

Figure 5 plots the a summary of Tor's bandwidth distribution. The  $X$  axis is a given bandwidth, and the  $Y$  axis is the proportion of aggregated Tor's bandwidth less than  $X$ , compared with the total Tor's bandwidth. To calculate this measure, we sort valid running Tor routers by ascending bandwidth, and we have  $\{B_1, \dots, B_n\}$ . The  $y$  value is calculated by

$$y(B_i) = \frac{\sum_{B_j \leq B_i} B_j}{\sum B_j}. \quad (7)$$

The theoretical curve in Figure 5 is derived by obtaining Tor router data from the directory authority. For experiments, we ran the Tor client (downloading a file every 10 seconds) for 2 days (9/15/2007-9/17/2007) and recorded data for the circuits created. The middle nodes of these recorded paths can be used for deriving the figure by applying Formula (7). We can see that the theoretical curve and experimental curve match very well. Remark that Tor assumes that a bandwidth of more than  $1,500KB/s$  is not trustworthy and clips bandwidth to  $1,500KB/s$ , avoiding some attacks. We also used clipped bandwidth when evaluating the weighted bandwidth path selection algorithm.

<sup>4</sup>In the course of our experiments, we fixed a number of problems with the Tor source code (0.1.1.26) to be certain these options worked consistently.

From Figure 5, we can see that if one Tor router is chosen using the bandwidth weighted algorithm, there is a probability of 24.5% that a Tor router with a bandwidth smaller than 199KB/s is chosen. If two Tor routers are chosen using that algorithm, the probability will be  $1 - (1 - 0.245)^2 \approx 43.0\%$ . If three Tor routers are chosen that way, the probability will be  $1 - (1 - 0.245)^3 \approx 57.0\%$ . Thus the probability that a slow Tor router is chosen is large enough to make the overall Tor network slow. Since many TCP connections share the bandwidth, the performance degrades further.

Figure 6 shows the TCP throughput measured while downloading a file of size 458,752 bytes from a school web server over two pre-selected paths between 12:00AM - 3:00PM on September 10, 2007. The figure includes the names of those routers (all within USA) along the path, the mean value of TCP throughput for the path in Tor and the confidence interval (CI) for the measurement at a level of significance of  $\alpha=0.05$ . The sample size is 200. We chose the same country and late night for measurement in order to reduce the impact of possible factors such as cross traffic. We can draw two conclusions from Figure 6. First, although the expected TCP throughput along a path decreases as the path length increases, it is possible that particular instances of a long path may have better throughput than some particular short paths. In this case, the path of length 4 has a mean throughput of 42.22KB/s, greater than that of the 19.61KB/s achieved by the path of length 3. Second, **a router with small donated bandwidth anywhere on a path determines the path throughput**. The reason that the shorter path in Figure 6 has worse throughput is that the middle router (*illiithid*) of the short path has a low bandwidth.

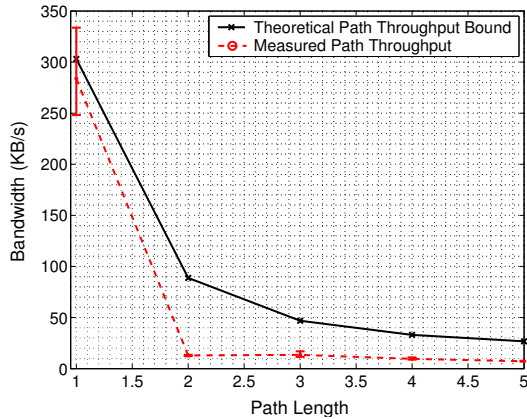


Fig. 7. Tor Path Throughput vs. Path Length

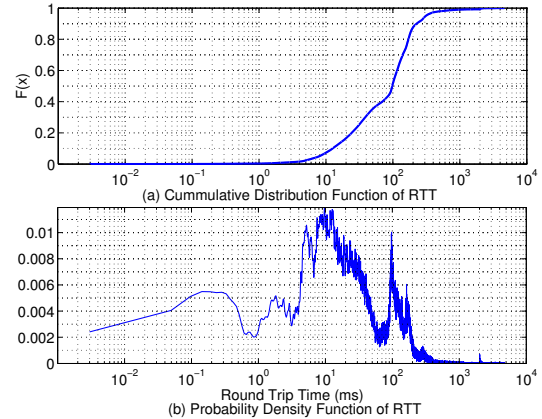


Fig. 8. RTT between Onion Routers

Figure 7 gives expected and measured Tor path throughput in terms of path length. For each path length, we downloaded a file every 10 minutes for one day and derived its rate. As Theorem 2 predicts, as the path length increases, the overall TCP throughput decreases. It can be observed that there is sharp drop when the path length increases from 1 to 2. This is because the probability of choosing a router with small bandwidth increases sharply as the path length increases from one to two. Beyond a path length of three, throughput decreases more slowly. In order to increase anonymity, we may increase the Tor path length beyond three and the performance will not degrade much beyond performance for a path of length three. From Figure 7, we can see that our theory matches well with the empirical results.

Figure 8 shows the distribution of round trip time (RTT) between Tor routers. We used the *King* approach [23] to measure the RTT<sup>5</sup>. From Figure 8, we can see that around 50% of measured round trip times are below 100ms. This is a relatively large value for TCP connections. Based on

<sup>5</sup>Because of the limitations of this method, RTT between some routers pairs cannot be measured.

Formula (1), the large round trip times reduce the TCP throughput for each segment of the Tor path. Hence, large round trip times contribute another reason for the Tor network to be less than might be hoped.

To summarize, based on the experimental results above, **we contribute three explanation for Tor's poor TCP performance.** (i) The key reason is that Tor selects path nodes in a probabilistic way, so that routers with low bandwidth have a high probability of being part of a path, and limiting the path bandwidth. As path lengths are increased to enhance anonymity, the probability that routers with low bandwidth are chosen for a path increases. Once any slow router is chosen, TCP throughput for a path will decrease dramatically. Tor's *hidden service* [14] may use a path of length 5 and the speed of such a service is often intolerable. (ii) Tor routers are distributed worldwide, creating the possibility of large round trip times between connected Tor routers and causing circuit segments built between them to have large round trip times and high packet drop rates. Both factors contribute to slow TCP connections based on Formula (1). (iii) Tor routers and links are shared by a large number of TCP connections and contention for the shared bandwidth effectively reduces each TCP connection's bandwidth, and hence the overall throughput for a path through the Tor network is lower.

## V. RELATED WORK

Since Chaum proposed the idea of anonymous computation and communication [1], researchers have applied these ideas in various ways, including message-based email systems and flow-based low-latency communications. Mix techniques can be used for either message-based (high-latency) or flow-based (low-latency) anonymity applications. Message-based email anonymity applications include the first Internet anonymity *remailer* by Helsingius [24] etc. Low-latency anonymous communication systems have two categories: *core mix* based networks and *P2P* based networks. *Tor* [4], *Onion routing* [25], *Freedom* [18] and many others belong to the first category. *Crowds* [26], *Tarzan* [27], *ANODR* [28] and many others belong to second category.

There is little systematic analysis of the QoS for anonymous communication networks. Rennhard *et al.* [29] empirically analyzed the performance of web browsing in a mix network using a synchronized dummy message generation scheme. Fu *et al.* [21] provided an improved version of this scheme and performed empirical analysis. McCoy *et al.* in [13] plainly presented some results of Tor's performance measurement including router geopolitical distributions, circuit latency and throughput. Fu *et al.* in [30] analyzed TCP performance in a flow-based mix network where batching and reordering schemes are introduced.

## VI. CONCLUSION

In this paper, we reported theoretical and empirical results from investigating why the popular anonymous communication network Tor may have poor TCP throughput. We analyzed the TCP throughput of Tor and found that marked TCP throughput degradation can be caused by a random path selection strategy (adopted for its value in preserving anonymity). Due to a high percentage of low-bandwidth routers on the Tor network, we find that the recently developed bandwidth weighted algorithm is limited in the performance improvements it provides. Both theoretical results and experimental data corroborate our findings. We also suggested potential alternatives which might improve performance on the Tor network.

Our data and conclusions should contribute to a deeper understanding of the root causes of performance shortfalls in the Tor network. These results may aid both researchers and developers working to understand and enhance the performance of Tor. Our future work includes developing appropriate metrics to support effective tradeoffs between anonymity and QoS within the Tor network. While the empirical data was collected from the Tor network, because it corroborates the theoretical analysis, the results of this work can be applied to other overlay networks adopting similar path selection schemes to provide anonymity.

## REFERENCES

- [1] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 4, no. 2, February 1981.
- [2] I. Goldberg, "Useful security and privacy," International Conference on Distributed Computing Systems (ICDCS) Panel: Looking Ahead: A Ten-Year Outlook for Internet Security and Privacy, 2007.
- [3] G. Danezis, R. Dingleline, and N. Mathewson, "Mixminion: Design of a Type III Anonymous Remailer Protocol," in *Proceedings of the 2003 IEEE Symposium on Security and Privacy (S&P)*, May 2003.
- [4] R. Dingleline, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *Proceedings of the 13th USENIX Security Symposium*, August 2004.
- [5] W. Yu, X. Fu, S. Graham, D. Xuan, and W. Zhao, "Dsss-based flow marking technique for invisible traceback," in *Proceedings of the 2007 IEEE Symposium on Security and Privacy (S&P)*, May 2007.
- [6] A. Serjantov, R. Dingleline, and P. Syverson, "From a trickle to a flood: active attacks on several mix types," in *Proceedings of Information Hiding Workshop (IH)*, February 2002.
- [7] X. Fu, Y. Zhu, B. Graham, R. Bettati, and W. Zhao, "On flow marking attacks in wireless anonymous communication networks," in *Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS)*, April 2005.
- [8] M. Wright, M. Adler, B. N. Levine, and C. Shields, "Defending anonymous communication against passive logging attacks," in *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, May 2003.
- [9] B. N. Levine, M. K. Reiter, C. Wang, and M. Wright, "Timing attacks in low-latency mix-based systems," in *Proceedings of Financial Cryptography (FC)*, February 2004.
- [10] S. J. Murdoch and G. Danezis, "Low-cost traffic analysis of tor," in *Proceedings of the IEEE Security and Privacy Symposium (S&P)*, May 2006.
- [11] L. Overlier and P. Syverson, "Locating hidden servers," in *Proceedings of the IEEE Security and Privacy Symposium (S&P)*, May 2006.
- [12] X. Wang, S. Chen, and S. Jajodia, "Network flow watermarking attack on low-latency anonymous communication systems," in *Proceedings of the IEEE Symposium on Security & Privacy (S&P)*, May 2007.
- [13] D. McCoy, K. Bauer, D. Grunwald, P. Tabriz, and D. Sicker, "Shining light in dark places: A study of anonymous network usage," University of Colorado at Boulder, Tech. Rep., August 2007.
- [14] R. Dingleline, N. Mathewson, and P. Syverson, "Tor: anonymity online," <http://tor.eff.org/index.html.en>, 2007.
- [15] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the Internet," *IEEE/ACM Transactions on Networking*, vol. 7, no. 4, pp. 458–472, 1999.
- [16] J. Padhye, V. Firoiu, D. Towsley, and J. Krusoe, "Modeling TCP throughput: A simple model and its empirical validation," in *Proceedings of ACM SIGCOMM*, September 1998.
- [17] Y. Liu, Y. Gu, H. Zhang, W. Gong, and D. Towsley, "Application level relay for high-bandwidth data transport," in *Proceedings of the 1-th International Workshop on Networks for Grid GridNets*, May 2004.
- [18] P. Boucher, A. Shostack, and I. Goldberg, "Freedom systems 2.0 architecture," December 2000.
- [19] R. Dingleline and N. Mathewson, "Tor path specification," <http://tor.eff.org/svn/trunk/doc/spec/path-spec.txt>, 2007.
- [20] K. Bauer, D. McCoy, D. Grunwald, T. Kohno, and D. Sicker, "Low-resource routing attacks against anonymous systems," University of Colorado at Boulder, Tech. Rep., August 2007.
- [21] Y. Zhu, X. Fu, B. Graham, R. Bettati, and W. Zhao, "On flow correlation attacks and countermeasures in mix networks," in *Proceedings of Workshop on Privacy Enhancing Technologies (PET)*, May 2004.
- [22] S. Chen and K. Nahrstedt, "An overview of quality-of-service routing for the next generation high-speed networks: problems and solutions," *IEEE Network*, vol. 12, no. 6, pp. 64–79, November 1998.
- [23] K. P. Gummadi, S. Saroiu, and S. D. Gribble, "King: Estimating latency between arbitrary internet end hosts," in *Proceedings of SIGCOMM Internet Measurement Workshop (IMW)*, August 2002.
- [24] J. Helsingius, "Press release: Johan helsingius closes his internet remailer," [http://www.eff.org/Censorship/Foreign\\_and\\_local/Finland/960830\\_penet\\_closure.announce](http://www.eff.org/Censorship/Foreign_and_local/Finland/960830_penet_closure.announce), August 1996.
- [25] P. F. Syverson, D. M. Goldschlag, and M. G. Reed, "Anonymous connections and onion routing," in *Proceedings of IEEE Symposium on Security and Privacy (S&P)*, May 1997.
- [26] M. Reiter and A. Rubin, "Crowds: Anonymity for web transactions," *ACM Transactions on Information and System Security*, vol. 1, no. 1, 1998.
- [27] M. J. Freedman and R. Morris, "Tarzan: a peer-to-peer anonymizing network layer," in *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS)*, November 2002.
- [28] J. J. Kong and X. Y. Hong, "ANODR: anonymous on demand routing with untraceable routes for mobile ad-hoc networks," in *Proceedings of the 4th ACM international symposium on Mobile Ad-Hoc Networking and Computing (MobiHoc)*, June 2003.
- [29] M. Rennhard, S. Rafaeli, L. Mathy, B. Plattner, and D. Hutchison, "Analysis of an anonymity network for web browsing," in *Proceedings of the IEEE the 11-th International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE)*, June 2002.
- [30] X. Fu, S. Jiang, W. Yu, S. Graham, and Y. Guan, "On tcp performance in flow-based mix networks," in *Proceedings of the 3th IEEE International Symposium on Dependable, Autonomic and Secure Computing (DASC)*, September 2007.

## APPENDIX A

In this appendix, we prove Theorem 1.

**Proof:**

Use  $\binom{n}{m}$  to multiply the left and right of (3), we have

$$\begin{aligned} \binom{n}{m} B(n, m) &= \binom{n-1}{m-1} B_n + \binom{n-2}{m-1} B_{n-1} + \cdots \\ &\quad + \binom{m}{m-1} B_{m+1} + \binom{m-1}{m-1} B_m. \end{aligned} \quad (8)$$

Since

$$\binom{n}{m} = \binom{n-1}{m} + \binom{n-1}{m-1}, \quad (9)$$

$$\binom{n-1}{m} = \binom{n-2}{m} + \binom{n-2}{m-1}, \quad (10)$$

$$\dots \quad (11)$$

$$\binom{m+1}{m} = \binom{m}{m} + \binom{m}{m-1}, \quad (12)$$

$$\binom{m}{m} = \binom{m-1}{m} + \binom{m-1}{m-1} \quad (13)$$

$$= 0 + \binom{m-1}{m-1} = \binom{m-1}{m-1}. \quad (14)$$

Substitute the last group of formulae into (8), we have

$$\begin{aligned} \binom{n}{m} B(n, m) &= \left( \binom{n}{m} - \binom{n-1}{m} \right) B_n + \left( \binom{n-1}{m} - \binom{n-2}{m} \right) B_{n-1} \\ &\quad + \cdots \\ &\quad + \left( \binom{m+1}{m} - \binom{m}{m} \right) B_{m+1} + \left( \binom{m}{m} - 0 \right) B_m \\ &= \binom{n}{m} B_n + \binom{n-1}{m} (B_{n-1} - B_n) + \binom{n-2}{m} (B_{n-2} - B_{n-1}) + \cdots \\ &\quad + \binom{m+1}{m} (B_{m+1} - B_m) + \binom{m}{m} (B_m - B_{m+1}) \\ &= \binom{n}{m} B_n + \sum_{i=1}^{n-m} \binom{n-i}{m} (B_{n-i} - B_{n-i+1}). \end{aligned}$$

Then divide the left and right of the above equation by  $\binom{n}{m}$ , we have

$$B(n, m) = B_n + \sum_{i=1}^{n-m} \frac{\binom{n-i}{m}}{\binom{n}{m}} (B_{n-i} - B_{n-i+1}). \quad (15)$$

Let

$$B_{n-1} - B_n = \delta_1, \quad (16)$$

$$B_{n-2} - B_{n-1} = \delta_2, \quad (17)$$

$$\dots \quad (18)$$

$$B_1 - B_2 = \delta_{n-1}. \quad (19)$$

Then

$$B(n, m) = B_n + \sum_{i=1}^{n-m} \frac{\binom{n-i}{m}}{\binom{n}{m}} \delta_i. \quad (20)$$

From (3), we know

$$B(n, m-1) = \frac{\sum_{k=m-1}^n \binom{k-1}{m-2} B_k}{\binom{n}{m-1}}. \quad (21)$$

Using the similar deduction of (20), we have

$$\begin{aligned} B(n, m-1) &= B_n + \sum_{i=1}^{n-m+1} \frac{\binom{n-i}{m-1}}{\binom{n}{m-1}} \delta_i \\ &= B_n + \sum_{i=1}^{n-m} \frac{\binom{n-i}{m-1}}{\binom{n}{m-1}} \delta_i + \frac{1}{\binom{n}{m-1}} \delta_{n-m+1}. \end{aligned} \quad (22)$$

Then the ratio of  $\delta_i$  in (22) and (20) is

$$\frac{\binom{n-i}{m-1}}{\binom{n}{m-1}} / \frac{\binom{n-i}{m}}{\binom{n}{m}} = \frac{n-m+1}{n-i-m+1} \quad (23)$$

$$\geq 1. \quad (24)$$

Then we have

$$\frac{\binom{n-i}{m-1}}{\binom{n}{m-1}} \geq \frac{\binom{n-i}{m}}{\binom{n}{m}}. \quad (25)$$

Therefore

$$B(n, m-1) - B(n, m) = \sum_{i=1}^{n-m} \left( \frac{\binom{n-i}{m-1}}{\binom{n}{m-1}} - \frac{\binom{n-i}{m}}{\binom{n}{m}} \right) \delta_i + \frac{1}{\binom{n}{m-1}} \delta_{n-m+1}.$$

Since  $\delta_i > 0$ , we have

$$B(n, m-1) - B(n, m) \geq 0 \quad (26)$$

$$B(n, m-1) \geq B(n, m). \quad (27)$$

■

## APPENDIX B

In this appendix, we prove Theorem 2.

**Proof:**

From (3), we know

$$B(l, m) = \frac{\sum_{k=m}^l \binom{k-1}{m-1} B_k}{\binom{l}{m}} = \frac{\sum_{k=m}^l \binom{n}{m} \binom{k-1}{m-1} B_k}{\binom{l}{m} \binom{n}{m}}. \quad (28)$$

and

$$B(n, m) = \frac{\sum_{k=m}^n \binom{k-1}{m-1} B_k}{\binom{n}{m}} \quad (29)$$

$$= \frac{\sum_{k=l+1}^n \binom{k-1}{m-1} B_k}{\binom{n}{m}} + \frac{\sum_{k=m}^l \binom{k-1}{m-1} B_k}{\binom{n}{m}} \quad (30)$$

$$= \frac{\sum_{k=l+1}^n \binom{l}{m} \binom{k-1}{m-1} B_k}{\binom{l}{m} \binom{n}{m}} + \frac{\sum_{k=m}^l \binom{l}{m} \binom{k-1}{m-1} B_k}{\binom{l}{m} \binom{n}{m}}. \quad (31)$$

Then

$$\begin{aligned} B(l, m) - B(n, m) &= \frac{\sum_{k=m}^l \binom{n}{m} \binom{k-1}{m-1} B_k}{\binom{l}{m} \binom{n}{m}} - \frac{\sum_{k=m}^l \binom{l}{m} \binom{k-1}{m-1} B_k}{\binom{l}{m} \binom{n}{m}} - \frac{\sum_{k=l+1}^n \binom{l}{m} \binom{k-1}{m-1} B_k}{\binom{l}{m} \binom{n}{m}} \\ &= \frac{\sum_{k=m}^l \left( \binom{n}{m} \binom{k-1}{m-1} - \binom{l}{m} \binom{k-1}{m-1} \right) B_k}{\binom{l}{m} \binom{n}{m}} - \frac{\sum_{k=l+1}^n \binom{l}{m} \binom{k-1}{m-1} B_k}{\binom{l}{m} \binom{n}{m}} \\ &> B_l \frac{\sum_{k=m}^l \left( \binom{n}{m} \binom{k-1}{m-1} - \binom{l}{m} \binom{k-1}{m-1} \right)}{\binom{l}{m} \binom{n}{m}} - B_l \frac{\sum_{k=l+1}^n \binom{l}{m} \binom{k-1}{m-1}}{\binom{l}{m} \binom{n}{m}} \\ &> B_l \frac{\binom{n}{m} \sum_{k=m}^l \binom{k-1}{m-1} - \binom{l}{m} \sum_{k=m}^l \binom{k-1}{m-1}}{\binom{l}{m} \binom{n}{m}}. \end{aligned} \quad (32)$$

From the knowledge of Combinatorics, we know

$$\sum_{k=m}^l \binom{k-1}{m-1} = \binom{l}{m}, \quad (33)$$

$$\sum_{k=m}^n \binom{k-1}{m-1} = \binom{n}{m}. \quad (34)$$

Substitute (33) and (34) into (32),

$$B(l, m) - B(n, m) > B_l \frac{\binom{n}{m} \binom{l}{m} - \binom{l}{m} \binom{n}{m}}{\binom{l}{m} \binom{n}{m}} \quad (35)$$

$$> 0. \quad (36)$$

Therefore

$$B(l, m) > B(n, m). \quad (37)$$

■