Enhancing Tor Performance For Bandwidth-Intensive Applications†

Technical Report BUCS-TR-2012-013

RAYMOND SWEHA
remos@cs.bu.edu
Computer Science Dept
Boston University, USA

AZER BESTAVROS
best@cs.bu.edu
Computer Science Dept
Boston University, USA

IBRAHIM MATTA
matta@cs.bu.edu
Computer Science Dept
Boston University, USA

***Abstract*—When it was first introduced a decade ago, Tor, the anonymous onion routing protocol, aimed at providing anonymity for latency-sensitive applications, such as web-browsing, as opposed to bandwidth-intensive applications, such as on-demand or live video streaming. This emphasis on latency-sensitive applications is evident from proposed Tor circuit-scheduling techniques [23], [10] that throttle bandwidth-intensive applications in favor of bursty, latency-sensitive applications. In this paper, we deviate from this traditional view by identifying key attributes and design decisions that negatively impact Tor's performance in general and its ability to cater to bandwidth-intensive applications in particular, and by proposing new capabilities that aim to enhance Tor's performance as it relates to anonymizing bandwidth-intensive traffic. We present results from in-vivo measurement studies that shed light on Tor's approach to manage load across relays, which manifests itself in the way source-based routing at the end-systems (clients) is handled. We present an analytical model that captures the key attributes of the feedback control inherent in Tor's approach to load management – namely, probing and circuit selection. We show that changing some of these key attributes yields measurable improvement in terms of overall network utilization as well as better load balancing of relays, resulting in better predictability of individual circuit performance. To boost the performance of bandwidth-intensive circuits, we propose the use of on-demand relays (angels) to not only increase the capacity in the Tor network, but also to implement special bandwidth-boosting functionality using multi-path routing. Our conclusions are backed up with results from simulation experiments.**

I. INTRODUCTION

Tor, a second generation onion routing system [1], is based on a source routing protocol that provides anonymity to its clients by routing traffic through a randomly-constructed circuit of intermediate relays. A client does not have to trust any node except the directory server from which it gets the list of relays that are presumed as independent and non-colluding. Typically, the client chooses three relays from such a list to construct a circuit (an overlay path) for traffic to/from any Internet host (server). Through proper encapsulation and encryption, traffic flows from one relay to the next in such a way that any node (relay) in the network can only glean a partial view of the overlay path between the client and the server, thus breaking any association between the source and destination of flows traversing the Tor network (and also hiding the identity of the client from the server). Although Tor is the most widely used anonymity network, it still suffers from major performance issues [10]. Tor developers and the research community are aware of these performance limitations, as evidenced by a plethora of studies and proposals aiming to enhance the performance of Tor [17], [23], [13], [11]. The main performance issue considered in these works is the highly unpredictable performance of circuits established through the Tor network, *e.g.*, resulting in poor HTTP response time. There have been two main hypotheses as to the culprit: (1) delay through a relay – namely, the amount of time a Tor packet (cell) spends in a Tor relay from the moment it is received till the moment it is sent to the next relay, and (2) latency of an overlay link: the time it takes the cell to traverse an overlay link, from the moment it is sent from one relay till the time it appears at the other end. For example, the authors of *Tunable Tor* [20] assumed relay delays are dominant, thus choosing relays based on their estimated throughput, whereas Sherr *et al.* assumed overlay link latencies are dominant, thus choosing relays to minimize the sum of overlay link latencies.

Motivation and Scope: When it was first introduced a decade ago, Tor aimed at providing anonymity for latency-sensitive applications, such as web-browsing, as opposed to bandwidth-intensive applications, such as on-demand or live video streaming – hence the emphasis in the aforementioned studies on delays as the primary gauge of performance. In light of the increasing prevalence of streaming, in this paper we deviate from this traditional view by focusing on Tor's ability to deliver higher bit rates, consistently. At a very basic level, this is a matter of provisioning: adding relay "capacity" to the Tor network to allow it to move more bits-per-second. In prior projects of ours [22], [21], we addressed similar provisioning problems in peer-to-peer (P2P) overlays through the use of *angels* – special resources acquired on demand (*e.g.*, from the cloud) for the sole purpose of improving the fidelity of a service. Adding raw capacity to make up a deficit in a particular resource is not a panacea. In addition to deploying angels (if needed), it is necessary to ensure that the underlying P2P system is able to optimally capitalize on this added capacity. To do so, it is often necessary to make judicious decisions, not only as it relates to the number, capacity, and placement of angels, but also as it relates to altering the way angels participate in the underlying P2P protocol.

Paper Outline: In this paper, we evaluate the extent to which angels may be used in support of bandwidth-intensive applications in a Tor network: we examine the key attributes and design decisions that negatively impact Tor's performance

†This research was supported in part by NSF awards #0735974, #0820138, #0963974, #1012798, and by a Google 2011 Faculty Research award.

in general and its ability to cater to bandwidth-intensive applications in particular, and we propose new capabilities, supported through the use of angels, to enhance Tor's performance relating to anonymizing bandwidth-intensive traffic.

We start in Section II with some background on Tor's operation as it relates to how clients construct circuits in a way that is meant to achieve high utilization of Tor resources. We also review prior work aiming to improve Tor's performance. In Section III we present results from in-vivo measurement studies that we conducted to shed light on the effectiveness of the various mechanisms used for load distribution in Tor. Our first finding is that the primary culprit for Tor's poor performance is the uneven throughput of the relays that make up a circuit. This finding implies that better and more predictable performance would result if Tor's relays are load balanced, *i.e.*, the attainable throughput from individual relays is more even. Indeed, our second finding is that as currently implemented and configured, Tor results in an inadequately-balanced load distribution over relays, resulting in lower overall utilization and higher variance in circuit throughput – both of which are undesirable attributes.

Load distribution in the Tor network is achieved through a set of interacting mechanisms: a *probing mechanism* to estimate the available capacity of relays, a *feedback mechanism* to compute a normalized value that corresponds to the total capacity of the relay, and a *circuit construction mechanism* to preferentially assign load to underutilized relays. To gain insights into how the interaction between these processes affects load balancing, in Section IV, we present an analytical model that captures the key probing and circuit construction attributes of the feedback control inherent in Tor's load distribution mechanisms. Using this model, we show that changing some of these key attributes yields measurable improvement in terms of overall network utilization as well as lower variability of individual circuit performance. In Section V, we show that these improvements are tangible by presenting results from simulation experiments, in which assumptions are relaxed, and various variant mechanisms are evaluated.

While adding angels to a well-balanced and highly utilized Tor network ensures that the added capacity will be utilized judiciously, it does not provide us with a mechanism via which bandwidth-intensive applications are properly provisioned. In Section VI, we introduce a simple angel functionality that boosts the performance of bandwidth-intensive circuits using multi-path routing.

II. Background and Related Work

Tor Protocol Basics: Tor [1] is a source routing protocol that provides anonymity to its clients by means of routing their traffic through an encrypted circuit of intermediate relays (muxes). Tor relies on a core set of around $2,900$ persistent relays, typically operated by research institutes or privacy activists. Around $400,000$ clients make use of the Tor network, daily, with around $120,000$ active at any point in time [2]. Tor leverages the (processing and communication) resources of clients by allowing them to function as relays, boosting the overall capacity of the Tor network, which can be seen as a P2P overlay. A Tor client contacts a *directory server* to get a list of Tor relays, from which it constructs circuits (overlay paths). A relay in a circuit cannot identify other nodes in the circuit except for the relay that precedes it and the one that follows it. When a client wants to send a message (*e.g.*, an HTTP request), it recursively encrypts the message with symmetric keys shared with the relays in the circuit, starting with the exit relay (the relay that sends the HTTP request to, and receives the response from the web server). In the forward direction, each relay peels a layer of encryption (a layer of the onion), sending the resulting message along the path towards the exit relay. In the reverse direction, each relay, starting with the exit relay, adds an onion layer of encryption, sending the resulting message along the reverse path towards the client. Upon receiving such a message, the client recursively peels all layers and consumes the content (displays a web page). By construction, each byte of data sent or received by a client needs to traverse at least three relays, consuming communication and processing capacities at each.

Acquisition and Management of Relay Capacities in Tor: Given documented, chronic performance issues, an important aspect of the Tor protocol (and the subject of much research) relates to how Tor manages the capacity and utilization of various relays. In this paper, we use *relay capacity* to refer to the number of bytes that a relay can forward per second on all circuits going through it; we use *relay utilization* to mean the fraction of the relay capacity that is in use at any point in time; we use *normalized relay capacity* to refer to the relative overall capacity of a relay; and we use *relay available capacity* to mean the throughput that a circuit is able to command from the relay at any point in time, which is a function of the (raw) relay capacity as well as the number of circuits using that relay in a max-min fair fashion.

As we alluded before, Tor's load/resource management is done through three interacting mechanisms: a *probing mechanism*, a *feedback mechanism*, and a *circuit construction mechanism*. Tor's circuit construction mechanism seeks to improve the overall system utilization by increasing the selection probability of relays deemed to have higher capacities. In earlier versions of Tor, each relay self-reported its capacity. To protect against untruthful clients, the current implementation of Tor uses directory authorities (DAs) to estimate the normalized relay capacities [14]. Every hour, based on previous estimates, each DA partitions the list of relays into groups of 50, and constructs two-relay circuits using relays in the same group to download a file for the purpose of estimating the available capacity of the constituent relays. These available capacity estimates are fed to a PID controller [14] that implements the *feedback mechanism* for estimating the normalized relay capacities. Let F_i be the average throughput of the DA's relay-pair probes involving relay i. $F_i\ \forall i$ is fed into a PID controller [14], which produces an estimate of the normalized capacity for each relay at time t as follows:

$$EST_i^t = EST_i^{t-1} * \frac{F_i}{\sum_{\forall j \in R}\{F_j\}/|R|} \qquad \forall i \in R$$

The controller increases (decreases) its estimate of a relay's normalized capacity when the probe throughput through that relay is above (below) average, thus encouraging (discouraging) clients to (from) using this higher-capacity (lower-capacity) relay in circuits they construct in the future.

When a client contacts a set of DAs, it gets from each

DA the list of relays, as well as estimates of their normalized capacities, EST_i, according to that DA. A client takes the average of those estimates, also called the *consensus capacity*. When a client constructs a new circuit, it chooses the constituent relays with probability proportional to the consensus capacity of the relays. Tor clients construct up to twelve circuits and alternate their usage.

Link and Relay Delays in Tor: Concerned with the performance of latency-sensitive applications, Dhungel *et al.* [9] conducted a measurement study to dissect the delay characteristics of Tor circuits. They showed that the delay of a circuit depends on two components: the delay through the relays, and the delay attributed to the latency of traversed overlay links. They concluded that while relay delays are the principal contributor of circuit delays, a sizable minority of circuits are dominated by overlay link delays. More importantly, they found no correlation between the delay introduced by a relay and its advertised or consensus capacity. Moreover, they showed that relay delay fluctuates over time, except for relays with very high capacity. They suggested that Tor's token-bucket scheduler – for multiplexing the use of a relay among all circuits going through it – needed more frequent replenishment, as circuits using less congested relays end up exhausting their quota of tokens, resulting in underutilization of these relays. Dhungel $et\ al.$ stopped short of proposing circuit construction algorithms.

Relay-based vs Link-based Circuit Construction: In an attempt to address Tor's performance issues, two classes of circuit construction algorithms emerged: (1) relay-based circuit construction, and (2) link-based circuit construction.

Under the hypothesis that selecting powerful relays results in better performance, the authors of *Tunable Tor* [20] propose having relays (as opposed to DAs) estimate the relay available capacity, and having the consensus capacity be calculated by one authority that distills the normalized capacity estimates of all relays using *EigenSpeed*, a principle component analysis estimator. To minimize overhead, a relay does not actively probe other relays, but rather uses the circuits routed through those relays to passively estimate their available capacities. Tunable Tor allows Tor clients to trade-off anonymity for enhanced performance. A parameter α governs this tradeoff by decreasing the probability of choosing a relay exponentially with the product of α times the rank of the relay with respect to available capacity. If α equals zero, all relays are chosen with equal probability, achieving maximum anonymity at the expense of performance; as α grows, the more powerful relays are chosen with an increasingly higher probability.

Under the hypothesis that circuit construction should avoid overlay links that are congested or suffer from long delays due BGP routing, Sherr *et al.* [18], [19] proposed using Vivaldi [7], a decentralized coordinate system, to measure the distances between different relays. The virtual coordinate of each relay is adjusted every time a new RTT measurement to another relay is available. Any client can measure the overall delay of a circuit by summing the delay of each link in the circuit. A circuit is chosen with probability exponentially proportional to a tuning parameter α times the rank of the circuit in the sorted list of all possible circuits based on the estimated circuit delay. Again, the parameter α tunes the tradeoff between performance and anonymity. Panchenko $et\ al.$ [15] advocate link-based measurement to enhance path selection in Tor. They propose two link-based measurements: (1) actively measuring RTTs of circuits, (2) passively estimating link delay using the circuits passing through it. They acknowledge that estimating link statistics is quadratic in the number of relays and suggest selecting relays based on average link-distance to all other relays. They also mention a hybrid approach that combines relay capacity and link delay; they suggest ranking circuits with probability inversely proportional to their aggregate link delay and directly proportional to the minimum relay capacity.

III. A Measurement Study on Live Tor

Experimental Setup: We used a client machine at Boston University (BU) to create $1,000$ circuits, one after the other, over the course of a few days. The client uses Tor to download files from an FTP server, located at BU as well. Both the client and the FTP server are machines with powerful CPUs and 1Gbps connections to the internet, ensuring that any bandwidth bottleneck is in the Tor network, not in the client, the server, or their links. The three relays – entry, intermediate, and exit relays – vary from a circuit to another, in accordance with Tor's circuit construction mechanism. Tor maintains multiple circuits (typically twelve) at a time and chooses the one to use at random. In our experiment, we limit the number of established circuits to one, by setting the parameter ``__DisablePredictedCircuits 1''. We mine the log files produced by Tor to get the ID and IP address of the constituent circuit relays. Using a GeoIP service [3], we find the longitude and latitude coordinates of the relay, from which we are able to calculate the geographical distance between adjacent relays in a circuit. Although the geo-distance between two relays does not directly correspond to the propagation delay between these relays, it could serve as a meaningful first-order approximation [12], [16]. In addition, we mine the consensus file when each circuit is established (hourly) to collect the consensus capacity estimate for each constituent relay. We use each of the constructed circuits to measure the average download time of a 3MB file and that of a one-byte file (both averaged over eight trials).

Results: The first question to consider is whether the delays introduced by circuit links (as estimated by the sum of the distances between adjacent relays in a circuit) is correlated with the circuit throughput. If strong correlation exists, link-based circuit selection would be justified. Figure 1-left shows the correlation between the sum of link-distances in the circuit (x-axis) and the average duration to download a 3MB file (y-axis). The Pearson product-moment correlation coefficient is found to be $r = 0.1243$.[1] This result suggests that there is a very weak (negligible) correlation between the throughput of a circuit and the aggregate link distances (delays) between its relays.

The second question to consider is whether the latency of a circuit is correlated to its throughput. We define *circuit latency* to be the time it takes to download a minimal-size

[1]The average duration is the file size (3MB) divided by the throughput. We plot duration against distance since any correlation between the two is likely to be linear, and hence possible to gauge using the r statistic.

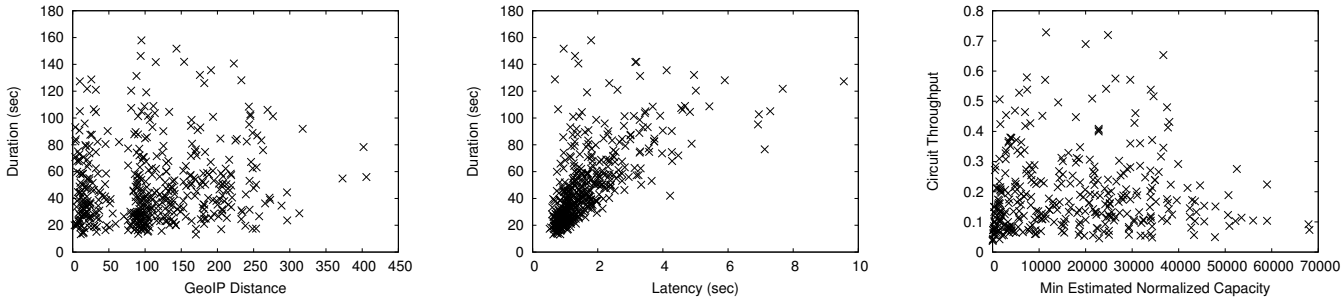

Fig. 1. Download time using a circuit shown on the y-axis versus the sum of its constituent link geo-distances (left), its latency, (middle) and the raw capacity of its constituent relays (right) shown on the x-axis.

(one-byte) file using that circuit. Clearly, latency captures the delays due to links (propagation delay) as well as relays (packet processing delays). Figure 1-middle shows the relationship between 3MB download times (as proxy for circuit throughput) on the y-axis and latencies on the x-axis. The correlation is evident as r equals 0.6553, suggesting that using latency as a first-order approximation of download duration is justifiable.

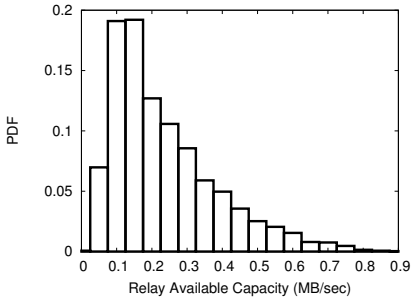

Fig. 2. PDF of relay available capacities, decaying exponentially.

The third question is whether the throughput of a circuit is correlated with the normalized capacities of the constituent relays. Figure 1-right shows the relationship between the throughput of a circuit in Mbps (x-axis) and the minimum of the consensus capacity estimates of the relays that make up the circuit (y-axis). The correlation is weak, as $r = 0.1496$, suggesting that Tor's three-pronged approach (using the probing, feedback control, and circuit construction mechanisms) to distribute load results in a good utilization across relays. In [24], the authors considered this lack of correlation as an evidence of failure. On the contrary, we argue that the lack of correlation is indicative of evenly utilized relays, which is one of the goals of the system.

The fourth question we consider is whether the clients' experience (the throughput observed by different clients or by the same client over time) is consistent. This is a fairness question, corresponding to a recurring complaint from Tor clients that sometimes they are stuck with low-throughput circuits [10]. Here we note that even though relays may be evenly utilized (as suggested by the results in Figure 1-right), it might be the case that the constructed circuits exhibit highly uneven performance. For the circuits we constructed, we noticed that the coefficient of variation (the standard deviation divided by the mean) of their achieved throughput is large – $CV = 0.62576$ – confirming that the fidelity (measured in terms of achievable bit rates) of Tor circuits is highly unpredictable.

The last question we consider is whether Tor's three-pronged approach to load distribution yields any particular probability distribution function (PDF) of relay available capacities. Recall that the available capacity of a Tor relay is the expected throughput of a circuit traversing that relay. For a fully-utilized relay, the available capacity is nothing other than the max-min fair-share for a circuit going through that relay. For a relay that is not fully utilized, the available capacity is the expected throughput of a newly created circuit going through that relay. To characterize relay available capacities, we construct a special two-relay circuit where the exit node is our local BU Tor relay `BostonUCompSci`. The circuit downloads three files of size 3MB each from a powerful BU server to a powerful BU client. This set-up is our best attempt to make sure that the circuit bottleneck is the entry relay. The average throughput of the circuit is recorded for each of the 2,917 relays that were present in the Tor network at the time of the experiment. Figure 2 shows the PDF of relay available capacities. It suggests that most relays have small available capacities, and that the number of relays with higher available capacity decays at an exponential rate.[2]

Summary: Our live measurements on Tor lead us to the following conclusions: (1) the throughput of a Tor circuit can be assumed to be largely independent from the geographical link-distance between relays in that circuit; (2) the latency of a circuit and its throughput are inversely, strongly correlated; (3) the throughput of a circuit is largely independent of the overall (normalized) capacity of its relays, suggesting that Tor is relatively efficient in its use of resources by keeping an even utilization of relays; (4) the throughput of different circuits vary greatly, suggesting that some clients suffer significant performance degradation; and (5) the distribution of the relay available capacity can be approximated using an exponential distribution.

IV. Tor Circuit Throughput: Analytical Model

Assuming that relay available capacities follow a certain probability distribution,[3] can we characterize the probability mass function of the throughput of a circuit constructed as a result of Tor (current or proposed) circuit construction mechanisms?

[2]This is important as it allows us to assume that relay available capacities follow an exponential distribution in Section IV-A.

[3]We derive two models: the first is under the assumption (verified through measurement) that relay available capacities are exponentially distributed, whereas the second is under the assumption that they are uniformly distributed.

In this section, we develop an analytical model that allows us to obtain such characterization, and consequently quantify the performance implications from variants of Tor circuit construction mechanisms, as well as variants of Tor probing mechanism.

A. Variability of Tor Circuit Throughput

As previously defined, the available capacity of a relay is the expected throughput of a circuit traversing that relay. In this subsection, we assume that the available capacity of a relay follows an exponential distribution with parameter λ: $f(x) = exp(\lambda) = \lambda e^{-\lambda x}$. The throughput of a circuit is the minimum throughput of its three constituent relays. From the literature of order statistics [8], the minimum of three random variables is a random variable of density function:

$$f_{1:3}(x) = 3(1-F(x))^2 f(x) = 3(1-(1-e^{-\lambda x}))^2 * (\lambda e^{-\lambda x})$$
$$= 3\lambda e^{-3\lambda x} = exp(3\lambda)$$

which is itself an exponentially distributed variable with a parameter three times the one for the relay available capacity. Knowing that the mean and standard deviation equal $\frac{1}{3\lambda}$, the coefficient of variation (CV) would equal one. This is a surprising result, CV is independent from λ, which suggests that if we manage to decrease the skewness in the distribution of the relay available capacity, the variability experienced by clients (due to different circuit capacities) would stay the same. We attribute this result to the memoryless nature of the exponential distribution.

B. Effect of "Best-Of-K" Circuit Selection

Tor clients typically create between two and twelve circuits and choose the one to use at random. In this section, we ask the question what if each client uses the best (as opposed to a random) circuit from the set of circuits it created? Doing that would eliminate slow circuits, such as those observed by Dhungel in [9]. Since circuit throughput is an exponentially distributed random variable with parameter 3λ, order statistics tells us that the maximum of k such circuits is also a random variable with PDF:

$$f_{k:k}(x) = k(F(x))^{k-1} f(x) = k(1-e^{-3\lambda x})^{k-1} 3\lambda e^{-3\lambda x}$$

This random variable has an expected value:

$$\mu_k = \int_0^\infty x f_{k:k}(x)dx = \int_0^\infty xk(1-e^{-3\lambda x})^{k-1}3\lambda e^{-3\lambda x}dx$$
$$= \frac{1}{3\lambda}\int_0^\infty ky(1-e^{-y})^{k-1}e^{-y}dy = \frac{H_k}{3\lambda}$$

where $y = 3\lambda x$ and $H_k = \sum_{i=1}^{k}\frac{1}{i}$. See Appendix A for the detailed derivation. Similarly, we can compute the variance of the random variable governing the circuit throughput:

$$var_k(x) = \int_0^\infty (x-\mu_k)^2 f_{k:k}(x)dx$$
$$= \int_0^\infty (x-\frac{H_k}{3\lambda})^2 k(1-e^{-3\lambda x})^{k-1}3\lambda e^{-3\lambda x}dx$$
$$= \frac{1}{9\lambda^2}\int_0^\infty k(y-H_k)^2(1-e^{-y})^{k-1}e^{-y}dy = \frac{b_k}{\lambda^2}$$

where $y = 3\lambda x$ and $b_k = \frac{2}{9}\sum_{i=1}^{k}\frac{1}{i}H_i - \frac{H_k^2}{9}$. See Appendix B for the detailed derivation.

While the coefficient of variance $CV_k = \frac{\sqrt{var_k(x)}}{\mu_k} = \frac{\sqrt{b_k}}{H_k/3}$ is independent of λ, the skewness of the relay available capacity distribution, it does depend on k, the number of circuits from which the best is chosen. Figure 3 shows CV as a function of k. CV decreases with the increase in the number of circuits, k.

Figure 3 also shows CV as a function of k, under the assumption that the relay available capacities follow a uniform distribution $U(a,b)$. In this case,

$$\mu_k = b + (b-a)\sum_{i=0}^{k} C_i^k \frac{(-1)^{i+1}}{3i+1}$$

$$var_k(x) = (b-a)^2(2\sum_{i=0}^{k}[C_i^k\frac{(-1)^i}{3i+2}] - [\sum_{i=0}^{k} C_i^k\frac{(-1)^{i+1}}{3i+1}]^2)$$

See Appendix C for the detailed derivation. Again, the CV of circuit throughput decreases with k. This result suggests that choosing the best-of-k circuit is not a mere byproduct of assuming an exponential distribution for the relay available capacities, thus it is a recommended practice. We will verify this result through simulation in the next section.

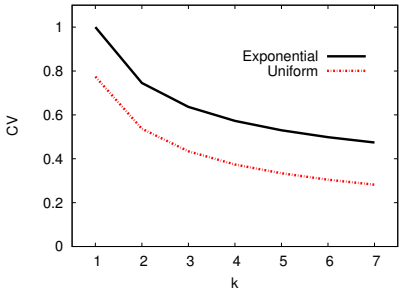

Fig. 3. The CV of the throughput of the best-of-k circuit.

C. Effect of Filtering out Probes with Below-Average Throughput on the Performance of the PID Controller

As we discussed earlier, the Directory Authority (DA) probes all relays regularly and feeds the PID controller with the average throughput of (typically five) probes as an estimate for the relay available capacity. The designers of the PID controller were concerned that some bad relay pairings would result in probes with very low throughput, lowering the estimate of available capacity, thus skewing the feedback signal to the PID controller. To mitigate this, the average of the five probes is calculated, and probes with below-average throughput are filtered out. The PID controller is fed with the *filtered* average probe throughput as the estimate for available capacity. This probe filtering mechanism turns out to have the inverse effect of what it was intended to achieve, resulting in an average estimate that is heavily skewed towards the maximum throughput (indeed, mostly equal to it).[4]

[4]Using our simulator, described in Section V, we noticed that typically, one of the five probes has a throughput that is significantly higher than the other probes. As a result, this maximum probe is the only one with throughput above the average probe throughput before filtering, effectively equating the filtered average with the maximum of the probes throughput.

To quantify the impact of probe filtering, we analytically derive the CV of the signal sent to PID controller with and without probe filtering. The estimate of the relay normalized capacity is $EST_i^t = EST_i^{t-1} * \frac{F_i}{\sum_{\forall j \in R}\{F_j\}/|R|} \quad \forall i \in R$, where F_i is a random variable representing the throughput of the average-of-l (no filtering) or max-of-l probe (filtering). A good controller would have the feedback signal: $s = \frac{F_i}{\sum_{\forall j \in R}\{F_j\}/|R|}$ approach one quickly. For a large set of relays, it is safe to assume that $\sum_{\forall j \in R}\{F_j\}/|R| \simeq E(F_i)$. The feedback signal is a random variable $s = \frac{F_i}{E(F_i)}$ with a mean of one and a standard deviation of $\frac{\sigma(F_i)}{E(F_i)}$. Hence, $CV = \frac{\sigma(F_i)}{E(F_i)}$. In the case of filtering, F_i is the max-of-l, which is mathematically identical to the derivation of best-of-k circuit selection in the previous section. In this case $CV = \sqrt{\sum_{i=1}^{l} \frac{2}{i} H_i - H_l^2} * \frac{1}{H_l}$, where $H_l = \sum_{i=1}^{l} \frac{1}{i}$. In the case of no filtering, F_i is the average-of-l probes, each is exponentially distributed, which is an Erlang-l distribution divided by l with a mean of $\frac{1}{2\lambda}$ and standard deviation of $\frac{\sqrt{l}}{l*2\lambda}$. Hence $CV = \frac{1}{\sqrt{l}}$.

Figure 4 plots the ratio between the CV of the signal sent to the PID controller with and without filtering. It suggests that the current version of Tor, with $l = 5$, feeds the PID controller a feedback signal that is 18.5% noisier (with higher variability) than our proposed version without filtering. Such a noisy feedback signal would negatively impact the normalized relay capacity estimates produced by the PID controller and, consequently, compromising the efficiency of the whole system.

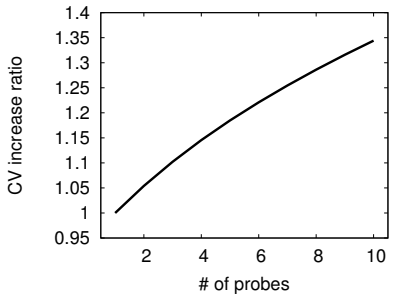

Fig. 4. The ratio between the CV of the signal sent the PID controller with filtering and without filtering as a function in the number of probes.

V. Experimental Evaluation

To confirm the conclusions from the analytical models in Section IV under relaxed assumptions (and additional parameters), to capture the feedback dynamics between Tor's probing, feedback control, and circuit construction mechanisms, and to evaluate our proposed improvements to these mechanisms, we built our own Tor simulator. In this section, after describing our simulator, we show a representative set of the experiments we conducted that give more insights into the behavior of Tor, and confirm our findings.

A. The Simulator

Our simulation model consists of three modules. The first module mimics the behavior of a large number of clients. Each client constructs a circuit, using three relays out of the available relays. The second module assigns throughput to constructed circuits given max-min fairness[5] and estimates relay available capacities, which determines the throughput available to a new circuit (probe). The third module models Tor's measurement infrastructure, which regularly probes all the relays, and uses PID control to update the normalized capacity estimates, hourly.

These three modules greatly influence each other as constructed circuits affect the throughput of the measurement probes, which in turn are used by Tor's PID controller to estimate each relay's normalized capacity, to be used by clients to bias their relay selection for circuits constructed in the next hour. Thus, over time, the simulator captures the change in relay capacity estimates and its effect on the achieved throughput of the constructed circuits. In particular, we adopt a quasi-static discrete-time simulation model wherein probing is performed at each time step (representing, say, 12 minutes), and weights for circuit selection are updated every $n > 1$ steps. We take $n = 5$ (representing an hour), and we assume that the system reaches steady state between time steps so circuits achieve their steady-state max-min fair share.

Circuits Construction: To model the circuits selected by Tor clients at a certain point in time, we create a certain number of circuits M at each time step, with a default of $M = 120,000$. We choose a circuit's constituent relays proportionally to their PID-provided capacity estimates as described in Section II. To simulate our modification of choosing the best-of-k circuit, a client builds k candidate circuits, compares their anticipated throughput, and chooses the best, with k between one and six. Because the selection of the best circuit depends on observed performance of already established circuits, while constructing the M circuits, we re-estimate the relay available capacities, and hence throughput of established circuits, every $m < M$ circuits, with a default of $m = M/10$. As noted earlier, the throughput of circuits is calculated assuming a max-min fair capacity allocation.

The PID Controller: As described in Section II, every hour, Tor probes each relay a number of times, at least five by default. Those probes provide the PID controller with an estimate of each relay's available capacity, *i.e.*, throughput of a circuit passing through the relay. If the average probe throughput for a relay is higher (lower) than the average for all relays, this indicates that this relay is relatively under-utilized (over-utilized), and the PID controller increases (decreases) its estimate of the relay normalized capacity and hence the relay's selection weight, enticing more circuits to (not) use this relay.

As noted earlier, we simulate probing at each time step. The probes are conducted over two-relay circuits. A probe throughput is computed in the same way as a circuit throughput, based on max-min capacity allocation. After probing all relays, we apply a filtering technique to drop some (or none) of each relay's probes. Tor's default behavior is to drop probes that have throughput that is below the average of all probes. Besides Tor's default behavior, our simulator supports: (1) *no filtering*, *i.e.*, use all probes; (2) drop the probe with the maximum throughput for each relay; or (3) drop the minimum and the maximum throughput probes for each relay. The PID updates its estimate of each relay normalized capacity (weight) and publishes it to clients (cf. Section II).

[5]Max-min fairness is typical of how resources are shared on the Internet.

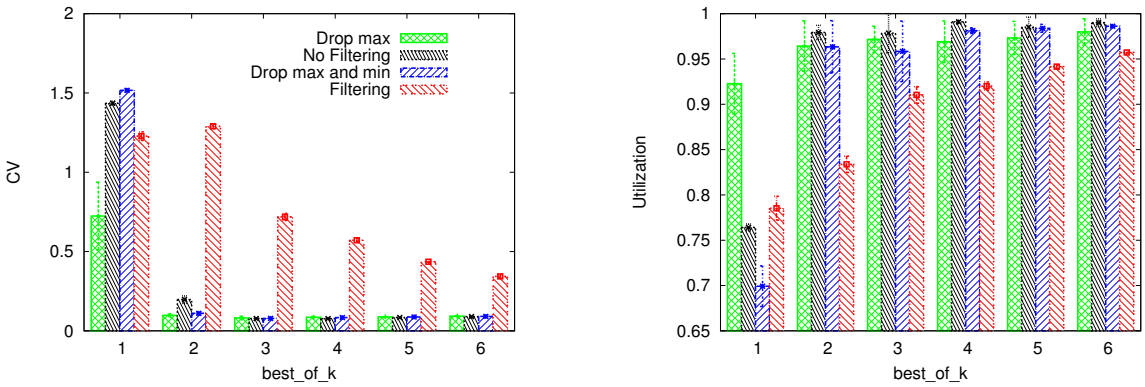

Fig. 5. The performance of different probe filtering technique against an increasing number of circuits from which the best is chosen.

To obtain performance metrics, each simulation runs for a duration of $25 \times n$ time steps, equivalent to updating circuit selection weights 25 times. Our simulation results follow.

B. Simulation Results

The base model for the following experiments mimics Tor's current default behavior. In it, a client constructs only one circuit and all the below-average probe values are filtered out. The number of probes is five. The relay capacities follows a bounded-Pareto distribution between 1,000 Kbps and 1,000,000 Kbps and a shape parameter $\alpha = 2$. We vary the number of relays (around 2,000 relays) while keeping the sum of the relay capacities (the aggregate capacity of the Tor network) equal to 3,900,000 Kbps. The number of circuits constructed is 120,000. Ideally, the capacity of relays would be assigned to the circuits allowing each one of them to have a throughput of around 35 Kbps. We set a limit on the throughput of each circuit that is uniformly distributed between 70 and 100 Kbps. This limit is introduced to mimic Tor's flow-control window mechanism [6], which limits the throughput of circuits to protect the Tor network from bandwidth-intensive clients like P2P applications.

The first experiment compares different probe filtering techniques against an increasing number of circuits k from which the client chooses the best. In Figure 5-left, the y-axis is the coefficient-of-variation (CV) of the constructed circuits, while the x-axis is the number of circuits from which the client chooses the best. When $k = 1$, the current Tor filtering technique (*i.e.*, drop probes that are below average) is outperformed by "drop max" but still beats the "no filtering" technique. For $k > 1$, "no filtering" outperforms other filtering techniques. Figure 5-right shows the aggregate utilization of relays (on the y-axis) with similar conclusions. Henceforth, the results are shown in a pair of graphs: one CV on the y-axis, and another utilization. All results show 95% confidence intervals.

For the same experiment, in Figure 6, we plot the PDF and CDF of the relay available capacities for two configurations: (1) vanilla Tor configuration with filtering deployed and $k = 1$, and (2) our proposed configuration with no filtering and $k = 6$. This result validates the model in Section IV in that the available relay capacity decays exponentially. It is worth noting though that vanilla Tor has a heavier tail, indicating that a few relays are left with a lot of available capacity, indicating inefficiency (bad load balancing).

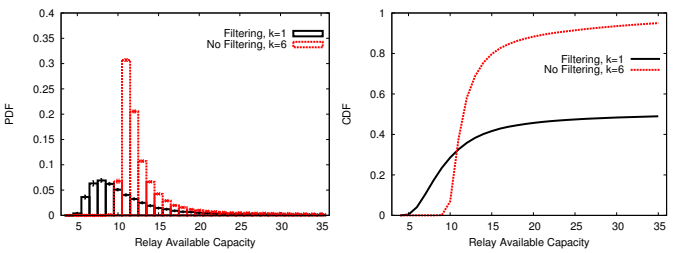

Fig. 6. The PDF and CDF of relay available capacities under two configurations: (1) Filtering, $k = 1$, and (2) No filtering, $k = 6$.

The following set of experiments examine the effect of changing certain parameters on Tor performance. The first is increasing the minimum number of probes that Tor conducts per relay. Figure 7 shows that as the number of probes increases, the utilization increases and the variation in client experience (circuit throughput) decreases. The improvement is limited though.

The next experiment studies the effect of changing the shape parameter α used to generate the relay capacities. It is important to note that we kept the aggregate capacity the same. Thus, a higher α results in fewer relays with more variable capacities. The higher variability of relay capacities results in a reduction in utilization (with wider confidence intervals) and also in more variability in the throughput of constructed circuits.

The next experiment considers the effect of increasing the aggregate capacity of the Tor network. The x-axis shows the factor by which the aggregate capacity of all relays is increased, *e.g.*, by a factor of 1, 1.2 and 1.4. The increased aggregate capacity translates into an increase in the number of relays, while the number of circuits remains the same. This does not seem to affect the utilization of relays but it lowers the CV of circuits, due to the decreased load on each relay (less circuits per relay).

Lastly, we study the effect of increasing the bound on the allowable circuit throughput. The average throughput is 35 Kbps, and we set the upper limit to be uniformly distributed U(70,70), U(70,100), U(70,140). As expected, allowing circuits to have higher throughput results in higher circuit throughput CV without a significant effect on the utilization of the relays.

VI. Bandwidth Boosting using Multi-Path Routing

So far, our effort has been to increase the utilization of the relays in the Tor network while limiting the unpredictability of clients' circuit throughput. In this section we propose enhancing the performance of certain clients using angels. Angels are special exit nodes with a lot of capacity. However, the

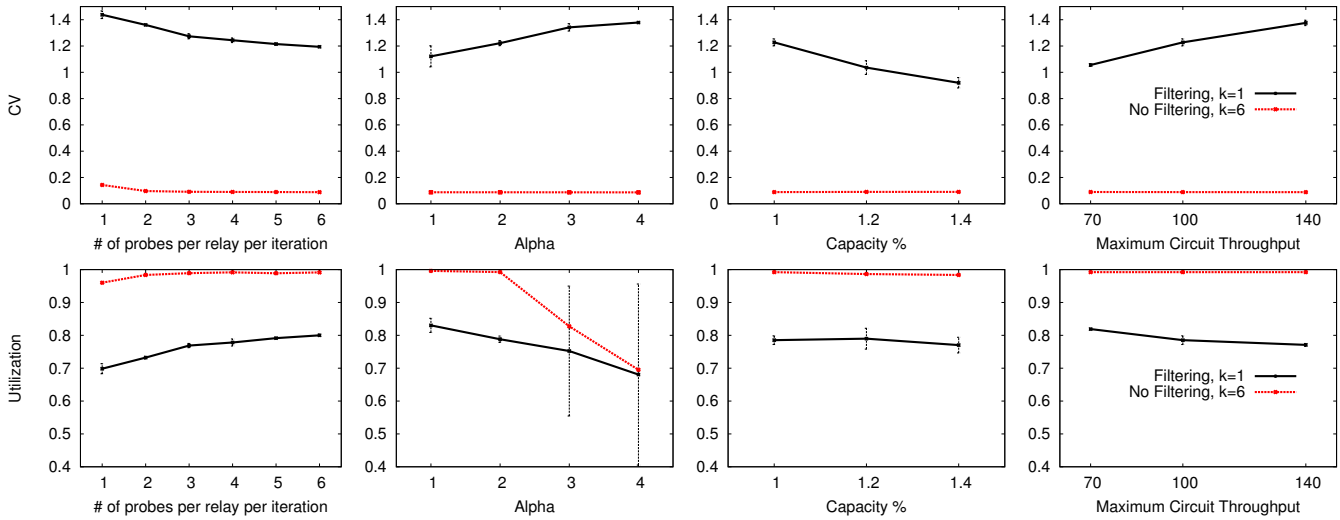

Fig. 7. The effects of various parameters on CV of circuit throughput (top) and utilization of relays (bottom).

throughput of a circuit is governed by the minimum available capacity of its constituent relays. To mitigate this problem, we adopt an approach originally proposed by AlSabah *et al.* [5], [4]. This approach uses multi-path forwarding to enhance the performance of constructed circuits. Figure 8 illustrates the idea. The angel (the exit node), is responsible for fetching content from, say, a web server and routing it back to the client. The angel can split the reply into segments, routing each through a different reverse path. The client receiving those segments reorders the received cells to reconstruct the original content (before being split by angels). For this to work, the client needs to actively construct those different paths and to inform the angel that those paths belong to the same circuit. This can be achieved through a nonce. When a path is extended to the angel, it will have a random nonce, if the angel has another concurrent circuit with the same nonce, it knows that both are paths of the same circuit. It is important to note that this capability doesn't require any modification to regular Tor relays (*i.e.*, relays that are not angels). Indeed, regular relays cannot even discern whether or not a circuit going through it is part of a multi-routed circuit. More importantly, angels performing the split functionality cannot glean any information pertaining to the identity of the client – only that some anonymous client wants to use a multi-path circuit. In this section, we show analytically and verify experimentally that angel-assisted multi-path circuit construction increases the throughput for clients and reduces the variance of constructed circuits. Analytically, we show that if all clients use multi-paths, the variance in their throughput will decrease.

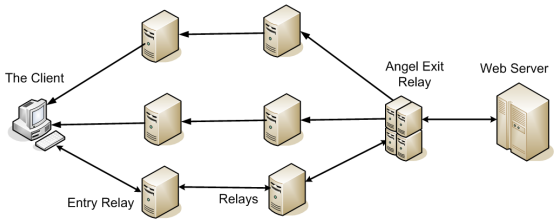

Fig. 8. An exit angel relay splits the response from the server to achieve better throughput using multiple circuits on the reverse path to the client.

We consider a multi-path enabled Tor network. Let there be n circuits at a certain point in time where each circuit has m-paths. Also, let the aggregate capacity of angels be C_a and the throughput of path i of circuit j be t_i^j. Consequently, the sum of the throughput of all paths of all circuits must not exceed the angel capacity, i.e., $\sum_{i=1}^{m}\sum_{j=1}^{n} t_i^j \leq C_a$. Assuming t_i^j follows an exponential distribution, then its parameter must be $\lambda = \frac{m*n}{C_a}$. Thus the throughput of a circuit is the sum of the throughput of its paths which is an Erlang distribution (sum of exponentials) with variance = $\frac{m}{\lambda^2} = \frac{C_a^2}{m*n^2}$. For vanilla Tor, *i.e.*, no multi-path routing, the sum of the throughput of all circuits must not exceed the angel capacity, *i.e.*, $\sum_{j=1}^{n} t^j \leq C_a$. Assuming t^j follows an exponential distribution, then its parameter must be $\lambda = \frac{n}{C_a}$, and its variance $\frac{1}{\lambda^2} = \frac{C_a^2}{n^2}$. This suggests that the variance in circuit throughput in case of multi-path construction is $\frac{1}{m}$ that of the vanilla Tor protocol.

The above analysis assumes that all clients use multi-path routing with the same number of paths. To evaluate the effect of deploying a limited number of angels, catering to a fraction of the traffic, we revert to simulation. We set up our simulator in such a way that 10% of the aggregate capacity of the network is provided through angels capable of the splitting functionality. Clients equipped with bandwidth-boosting capabilities inform angels acting as exit relays of the number of paths they require using special bit marking when setting up a circuit. In our simulation, this is set up as a uniformly random integer between one and five. We evaluate multi-path routing under three configurations: (1) "F, k=1" is Tor's default behavior, using probe filtering and best-of-1 circuit selection, (2) "NF, k=3" uses no filtering and best-of-3, and (3) "F, k=3" uses filtering and best-of-3.

Figure 9 is a scatter plot showing the results. On the x-axis is the number of paths a circuit is requesting and the y-axis is the average circuit throughput. The "NF, k=3" setting provides the highest utilization and the most predictable results. For example the relay utilization is 99% while the average circuit throughput when the number of paths is five is 4.99 (almost

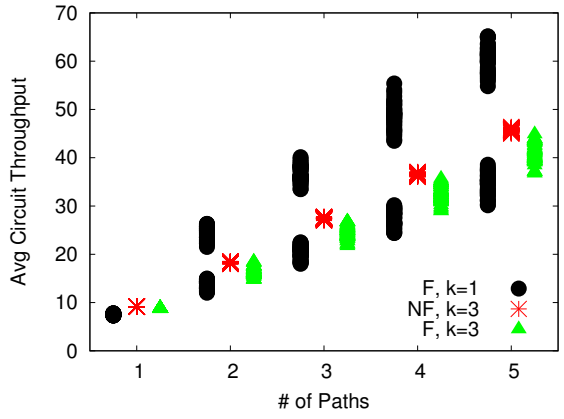

Fig. 9. The effects of splitting a circuit into multi-paths.

five) times the throughput of circuits with one path. The CV of constructed circuits is between 0.079 and 0.045 (near zero). These results should be contrasted to the "F, k=1" setting, where the relay utilization is 0.78 and the CV is between 1.2 and 0.6. The third "F, k=3" setting corresponds to the case when we have no influence on the directory authority (which uses filtering), with clients choosing the best-of-3 circuits, with multi-path forwarding enabled through angels. In this case the relay utilization is 0.91 and the CV is between 0.78 and 0.38. These results highlight the premise of multi-path angel-enabled routing as explained at the outset.

VII. Conclusion

In this paper, we set out to understand the behavior of Tor and how it could be possibly influenced to yield better, more predictable performance. We started with a characterization of throughput and delay performance for small and large data transfers, which we correlated to a number of relay/circuit attributes. We concluded that the mechanisms that Tor employs to distribute load across relays achieve the basic goal of high relay utilization, but that they leave much to be desired in terms of predictability due to the high variability of circuit throughput.

Armed with these observations, we used order statistics to analyze the coefficient-of-variation (CV) of throughput achieved by circuits, under Tor's default random circuit selection versus our proposed best-of-k strategy, and also under Tor's approach to probe filtering (of relay available capacity measurements) versus other variant probe filtering strategies, including no-filtering. We analytically showed a lower CV under the no-filtering and best-of-k approaches. We then validated this using a simulation model that captures the feedback dynamics between the processes responsible for measuring/estimating relay available/normalized capacities, and the circuit construction mechanism used by clients. To further improve performance, we proposed and evaluated the deployment of special relays (angels) capable of supporting multi-path routing of some client circuits. We show that even without disabling Tor's default filtering, we can achieve lower CV, and scalable higher throughput to multi-path capable, throughput-sensitive clients.

Our on-going work incorporates these improvements into the Tor system, by modifying only angel relays and clients empowered to use multi-path circuits, and by assessing the effectiveness of these improvements in the presence of legacy relays and clients. Another research direction of ours is to evaluate the effectiveness of angels in mitigating timing attacks, which create congestion to de-anonymize their victims. By alleviating this congestion, angels have the potential of improve the anonymity characteristics of Tor clients.

Acknowledgment

We would like to thank Vatche Ishakian for helpful discussions and feedback on many aspects of this work.

References

[1] The Tor Project, Inc. www.torproject.org.
[2] Tor Live Measurements. https://metrics.torproject.org/.
[3] FreeGeoIP. http://freegeoip.net/.
[4] AlSabah, M., Bauer, K., Elahi, T., and Goldberg, I. The path less travelled: Overcoming tors bottlenecks with multipaths.
[5] AlSabah, M., Bauer, K., Elahi, T., and Goldberg, I. Tempura: Improved tor performance with multipath routing.
[6] AlSabah, M., Bauer, K., Goldberg, I., Grunwald, D., McCoy, D., Savage, S., and Voelker, G. Defenestrator: Throwing out windows in tor. In *Privacy Enhancing Technologies* (2011), Springer.
[7] Dabek, F., Cox, R., Kaashoek, F., and Morris, R. Vivaldi: a decentralized network coordinate system. SIGCOMM '04, ACM, pp. 15–26.
[8] David, H., and Nagaraja, H. *Order statistics*. Wiley Online Library, 1970.
[9] Dhungel, P., Steiner, M., Rimac, I., Hilt, V., and Ross, K. Waiting for anonymity: Understanding delays in the tor overlay. In *P2P* (2010), IEEE, pp. 1–4.
[10] Dingledine, R., and Murdoch, S. Performance improvements on tor or, why tor is slow and what were going to do about it. *Online: http://www. torproject. org/press/presskit/2009-03-11-performance. pdf* (2009).
[11] He, J., Suchara, M., Bresler, M., Rexford, J., and Chiang, M. Rethinking internet traffic management: From multiple decompositions to a practical protocol. In *CoNEXT 2007*.
[12] Huffaker, B., Fomenkov, M., Plummer, D., Moore, D., and Claffy, K. Distance metrics in the internet. In *Proc. of IEEE International Telecommunications Symposium (ITS)* (2002).
[13] Jansen, R., Hopper, N., and Kim, Y. Recruiting new tor relays with braids. In *Proceedings of the 17th ACM conference on Computer and communications security* (2010), ACM, pp. 319–328.
[14] Loesing, K., Perry, M., and Gibson, A. Bandwidth Scanner Specification, 2011. https://gitweb.torproject.org/torflow.git/blob_plain/HEAD:/NetworkScanners/BwAuthority/README.spec.txt.
[15] Panchenko, A., and Renner, J. Path selection metrics for performance-improved onion routing. In *SAINT'09* (2009), IEEE, pp. 114–120.
[16] Pastor-Satorras, R., and Vespignani, A. *Evolution and structure of the Internet: A statistical physics approach. Section 10.2*. 2007.
[17] Reardon, J., and Goldberg, I. Improving tor using a tcp-over-dtls tunnel. In *Proceedings of the 18th conference on USENIX security symposium* (2009), USENIX Association, pp. 119–134.
[18] Sherr, M., Blaze, M., and Loo, B. Scalable link-based relay selection for anonymous routing. In *Privacy Enhancing Technologies* (2009), Springer, pp. 73–93.
[19] Sherr, M., Loo, B., and Blaze, M. Towards application-aware anonymous routing. In *Proceedings of the 2nd USENIX workshop on Hot topics in security* (2007), USENIX Association, p. 4.
[20] Snader, R., and Borisov, N. Improving security and performance in the tor network through tunable path selection. *Dependable and Secure Computing, IEEE Transactions on*, 99 (2010), 1–1.
[21] Sweha, R., Ishakian, V., and Bestavros, A. Angels In The Cloud: A Peer-Assisted Bulk-Synchronous Content Distribution Service. In *CLOUD'2011* (Washington DC, USA, 2011).
[22] Sweha, R., Ishakian, V., and Bestavros, A. Angelcast: Cloud-based peer-assisted live streaming using optimized multi-tree construction. In *MMSys* (2012), ACM.
[23] Tang, C., and Goldberg, I. An improved algorithm for tor circuit scheduling. In *Proceedings of the 17th ACM conference on Computer and communications security* (2010), ACM, pp. 329–339.

[24] WANG, T., BAUER, K., FORERO, C., AND GOLDBERG, I. Congestion-aware path selection for tor? In *16th International Conference on Financial Cryptography and Data Security* (Feb 2012).

APPENDIX A
CALCULATE μ_k FOR EXPONENTIAL DISTRIBUTION

$$\begin{aligned}
\mu_k &= \frac{1}{3\lambda}\int_0^\infty y d(1-e^{-y})^k \\
&= \frac{1}{3\lambda}\int_0^1 -ln(1-z)dz^k \\
Where: z &= (1-e^{-y}) \\
\mu_k &= \frac{-1}{3\lambda}[ln(1-z)z^k]_0^1 - \int_0^1 z^k * \frac{-1}{1-z}dz \\
&= \frac{-1}{3\lambda}[ln(1-z)z^k]_0^1 - \int_0^1 * \frac{(1-z^k)-1}{1-z}dz \\
&= \frac{-1}{3\lambda}[ln(1-z)z^k]_0^1 - \int_0^1 \sum_{i=0}^{k-1} z^i dz + \int_0^1 \frac{1}{1-z}dz \\
&= \frac{-1}{3\lambda}[ln(1-z)z^k - \sum_{i=0}^{k-1}\frac{z^{i+1}}{i+1} - ln(1-z)]_0^1 \\
&= \frac{1}{3\lambda}\sum_{i=1}^{k}\frac{1}{i} + \frac{1}{3\lambda}\lim_{z\to 1}[(1-z^k)ln(1-z) \\
&= \frac{1}{3\lambda}\sum_{i=1}^{k}\frac{1}{i} + \frac{1}{3\lambda}\lim_{z\to 1}(\sum_{i=0}^{k-1} z^i)\frac{ln(1-z)}{(1-z)}
\end{aligned}$$

Using l'Hopital rule by differentiating the nominator and the denominator:

$$\begin{aligned}
\mu_k &= \frac{1}{3\lambda}\sum_{i=1}^{k}\frac{1}{i} + \frac{k}{3\lambda}\lim_{z\to 1}\frac{1/(1-z)}{-1/(1-z)^2} \\
&= \frac{1}{3\lambda}\sum_{i=1}^{k}\frac{1}{i} + \frac{k}{3\lambda}\lim_{z\to 1} -(1-z) \\
\mu_k &= \frac{1}{3\lambda}\sum_{i=1}^{k}\frac{1}{i} = \frac{H_k}{3\lambda}
\end{aligned}$$

APPENDIX B
CALCULATE $var_k(x)$ FOR EXPONENTIAL DISTRIBUTION

$$\begin{aligned}
var_k(x) &= \int_0^\infty (x-\mu_k)^2 f_{k:k}(x)dx \\
&= \int_0^\infty (x-\frac{H_k}{3\lambda})^2 k(1-e^{-3\lambda x})^{k-1}3\lambda e^{-3\lambda x}dx \\
&= \frac{1}{9\lambda^2}\int_0^\infty k(y-H_k)^2(1-e^{-y})^{k-1}e^{-y}dy \\
Where: y &= 3\lambda x \\
var_k(x) &= \frac{1}{9\lambda^2}\int_0^\infty (y-H_k)^2 d(1-e^{-y})^k \\
&= \frac{1}{9\lambda^2}\int_0^1 (-ln(1-z)-H_k)^2 dz^k \\
Where: z &= (1-e^{-y})
\end{aligned}$$

$$\begin{aligned}
var_k(x) &= \frac{1}{9\lambda^2}[(ln(1-z)+H_k)^2 z^k]_0^1 \\
&+ \frac{-1}{9\lambda^2}\int_0^1 z^k * 2(ln(1-z)+H_k) * \frac{-1}{1-z}dz \\
var_k(x) &= \frac{1}{9\lambda^2}[(ln(1-z)+H_k)^2 z^k]_0^1 \\
&+ \frac{-1}{9\lambda^2}\int_0^1 2(ln(1-z)+H_k) * \frac{(1-z^k)-1}{1-z}dz \\
var_k(x) &= \frac{1}{9\lambda^2}[(ln(1-z)+H_k)^2 z^k]_0^1 \\
&+ \frac{-1}{9\lambda^2}\int_0^1 2(ln(1-z)+H_k) * \frac{-1}{1-z}dz \\
&+ \frac{-1}{9\lambda^2}\int_0^1 2(ln(1-z)+H_k)\sum_{i=0}^{k-1} z^i dz \\
var_k(x) &= \frac{1}{9\lambda^2}[(ln(1-z)+H_k)^2 z^k]_0^1 \\
&+ \frac{-1}{9\lambda^2}\int_0^1 2(ln(1-z)+H_k) * dln(1-z) \\
&+ \frac{-1}{9\lambda^2}\int_0^1 2ln(1-z)\sum_{i=0}^{k-1} z^i dz + \frac{-1}{9\lambda^2}[2H_k\sum_{i=0}^{k-1}\frac{z^{i+1}}{i+1}]_0^1 \\
var_k(x) &= \frac{1}{9\lambda^2}[(ln^2(1-z)+2H_k ln(1-z)+H_k^2)z^k]_0^1 \\
&+ \frac{-1}{9\lambda^2}[ln^2(1-z)+2H_k ln(1-z)]_0^1 \\
&+ \frac{-1}{9\lambda^2}\sum_{i=0}^{k-1}\frac{2}{i+1}\int_0^1 (i+1)ln(1-z)z^i dz + \frac{-1}{9\lambda^2}[2H_k * H_k] \\
var_k(x) &= \frac{1}{9\lambda^2}[(ln^2(1-z)+2H_k ln(1-z))(z^k-1)]_0^1 + \frac{1}{9\lambda^2}[H_k^2] \\
&+ \frac{1}{9\lambda^2}\sum_{i=0}^{k-1}\frac{2}{i+1}H_{i+1} + \frac{-2H_k^2}{9\lambda^2} \\
var_k(x) &= \frac{1}{9\lambda^2}[(ln^2(1-z)+2H_k ln(1-z))(z^k-1)]_0^1 + \frac{1}{9\lambda^2}[H_k^2] \\
&+ \frac{1}{9\lambda^2}\sum_{i=0}^{k-1}\frac{2}{i+1}H_{i+1} + \frac{-2H_k^2}{9\lambda^2} \\
var_k(x) &= \frac{1}{9\lambda^2}\lim_{z\to 1}\frac{ln^2(1-z)+2H_k ln(1-z)}{1/(z-1)}\sum_{i=0}^{k-1} z^i \\
&+ \frac{2}{9\lambda^2}\sum_{i=1}^{k}\frac{1}{i}H_i + \frac{-H_k^2}{9\lambda^2} \\
var_k(x) &= \frac{k}{9\lambda^2}\lim_{z\to 1}[\frac{2ln(1-z)/-(1-z)}{-1/(z-1)^2} + 0] + \frac{2}{9\lambda^2}\sum_{i=1}^{k}\frac{1}{i}H_i + \frac{-H_k^2}{9\lambda^2} \\
var_k(x) &= \frac{2}{9\lambda^2}\sum_{i=1}^{k}\frac{1}{i}H_i - \frac{H_k^2}{9\lambda^2}
\end{aligned}$$

APPENDIX C
THE MODEL ASSUMING A UNIFORM DISTRIBUTION

In this appendix we assume that the relay available capacities follow a uniform distribution: $f(x) = u(a,b) = \frac{1}{b-a}$. The throughput of a circuit is the minimum throughput of its three constituent relays. The minimum of three random variables is

a random variable of density function:

$$\begin{aligned} f_{1:3}(x) &= 3(1-F(x))^2 f(x) = 3(1-\frac{x-a}{b-a})^2 * \frac{1}{b-a} \\ &= \frac{3}{(b-a)^3} * (b-x)^2 \\ F_{1:3}(x) &= \int_a^x f_{1:3}(x)dx = \int_a^x \frac{3}{(b-a)^3} * (x^2 - 2bx + b^2)dx \\ &= \frac{3}{(b-a)^3}[\frac{x^3}{3} - bx^2 + b^2 x]_a^x \\ &= \frac{x^3 - 3bx^2 + 3ba^2 - b^3 + b^3 - 3b^2 a + 3ba^2 - a^3}{(b-a)^3} \\ &= 1 - (\frac{b-x}{b-a})^3 \end{aligned}$$

The expected value of the circuit throughput will be:

$$\begin{aligned} \mu &= \int_a^b x f_{1:3}(x)dx = \int_a^b x\frac{3}{(b-a)^3} * (b-x)^2 dx \\ &= \int_a^b -xd\frac{(b-x)^3}{(b-a)^3} = \int_1^0 ((b-a)y^{1/3} - b)dy \\ &= \frac{3(b-a)}{4}[y^{4/3}]_1^0 - b[y]_1^0 = \frac{b}{4} + \frac{3a}{4} \end{aligned}$$

The variance:

$$\begin{aligned} var(x) &= \int_a^b (x-\mu_k)^2 f_{1:3}(x)dx \\ &= \frac{3}{(b-a)^3}\int_a^b (x - \frac{b}{4} - \frac{3a}{4})^2 (b-x)^2 dx \\ &= \frac{3}{(b-a)^3}\int_a^b ((x-b) + \frac{3(b-a)}{4})^2 (x-b)^2 d(x-b) \\ &= \frac{3}{(b-a)^3}\int_{a-b}^0 (y + \frac{3(b-a)}{4})^2 * y^2 dy \\ \text{where } y &= x-b \\ var(x) &= \frac{3}{(b-a)^3}\int_{a-b}^0 (y^4 + \frac{6(b-a)y^3}{4} + \frac{9(b-a)^2 y^2}{16})dy \\ &= \frac{3}{(b-a)^3}[\frac{y^5}{5} + \frac{6(b-a)y^4}{16} + \frac{9(b-a)^2 y^3}{16*3}]_{a-b}^0 \\ &= \frac{3}{(b-a)^3}[\frac{-(a-b)^5}{5} - \frac{6(b-a)(a-b)^4}{16} \\ &\quad -\frac{3(b-a)^2(a-b)^3}{16}] \\ &= \frac{3}{(b-a)^3}[\frac{(b-a)^5}{5} - \frac{3(b-a)^5}{8} + \frac{3(b-a)^5}{16}] \\ &= 3(b-a)^2[\frac{1}{5} - \frac{3}{8} + \frac{3}{16}] = \frac{3}{80}(b-a)^2 \end{aligned}$$

The coefficient of variation then equals:

$$CV_k = \frac{\sqrt{var(x)}}{\mu} = \frac{\sqrt{3/80}(b-a)}{b - 3/4(b-a)}$$

Which means that the CV increases with the increase of the width of the support $[a, b]$. Thus if angels can be used to decrease this support, the client experience would be more predictable.

In the case of best-of-k circuit selection.

$$f_{k:k}(x) = k(F(x))^{k-1} f(x) = k(1-(\frac{b-x}{b-a})^3)^{k-1}\frac{3(b-x)^2}{(b-a)^3}$$

The mean circuit throughput in this case would be:

$$\begin{aligned} \mu_k &= \int_a^b x f_{k:k}(x)dx = \int_a^b x * k(1-(\frac{b-x}{b-a})^3)^{k-1}\frac{3(b-x)^2}{(b-a)^3}dx \\ &= \int_a^b xd(1-(\frac{b-x}{b-a})^3)^k \\ &= [x(1-(\frac{b-x}{b-a})^3)^k]_a^b - \int_a^b (1-(\frac{b-x}{b-a})^3)^k dx \\ &= [b-0] + \int_1^0 (1-y^3)^k (b-a)dy \end{aligned}$$

Where $y = \frac{b-x}{b-a}$

$$\mu_k = b + (b-a)\int_1^0 (1-y^3)^k dy = b + (b-a) * a_k$$

Where

$$\begin{aligned} a_k &= \int_1^0 (1-y^3)^k dy \\ &= \int_1^0 \sum_{i=0}^k C_i^k (-y^3)^i dy \\ &= (-1)^i \sum_{i=0}^k C_i^k \int_1^0 (y^{3i})dy \\ &= (-1)^i \sum_{i=0}^k C_i^k [\frac{y^{3i+1}}{3i+1}]_1^0 \\ &= \sum_{i=0}^k C_i^k \frac{(-1)^{i+1}}{3i+1} \\ \mu_k &= b + (b-a)\sum_{i=0}^k C_i^k \frac{(-1)^{i+1}}{3i+1} \end{aligned}$$

As for the variance of the circuit throughput:

$$\begin{aligned} var_k(x) &= \int_a^b (x-\mu_k)^2 f_{k:k}(x)dx \\ &= \int_a^b (x-\mu_k)^2 * k(1-(\frac{b-x}{b-a})^3)^{k-1}\frac{3(b-x)^2}{(b-a)^3}dx \\ &= \int_a^b (x-\mu_k)^2 d(1-(\frac{b-x}{b-a})^3)^k \\ &= [(x-\mu_k)^2(1-(\frac{b-x}{b-a})^3)^k]_a^b \\ &\quad -\int_a^b (1-(\frac{b-x}{b-a})^3)^k (2(x-\mu_k))dx \\ &= (b-\mu_k)^2 \\ &\quad -\int_1^0 (1-y^3)^k (2(b-(b-a)y-\mu_k))(-(b-a))dy \end{aligned}$$

$$
\begin{aligned}
&= ((b-a)a_k)^2 - 2(b-a)^2 \int_1^0 (1-y^3)^k (y+a_k) dy \\
&= ((b-a)a_k)^2 - 2(b-a)^2 \int_1^0 (1-y^3)^k y dy \\
&\quad -2(b-a)^2 a_k \int_1^0 (1-y^3)^k dy \\
&= ((b-a)a_k)^2 - 2(b-a)^2 \int_1^0 y \sum_{i=0}^{k} C_i^k (-y^3)^i dy \\
&\quad -2(b-a)^2 a_k^2 \\
&= -((b-a)a_k)^2 - 2(b-a)^2 \sum_{i=0}^{k} (-1)^i \int_1^0 C_i^k y^{3i+1} dy \\
&= -((b-a)a_k)^2 - 2(b-a)^2 \sum_{i=0}^{k} (-1)^i [C_i^k \frac{y^{3i+2}}{3i+2}]_1^0 \\
&= -((b-a)a_k)^2 + 2(b-a)^2 \sum_{i=0}^{k} [C_i^k \frac{(-1)^i}{3i+2}] \\
&= (b-a)^2 (2 \sum_{i=0}^{k} [C_i^k \frac{(-1)^i}{3i+2}] - a_k^2) \\
&= (b-a)^2 * b_k
\end{aligned}
$$

Where

$$b_k = 2 \sum_{i=0}^{k} [C_i^k \frac{(-1)^i}{3i+2}] - a_k^2$$

The coefficient of variation in this case would be:

$$CV_k = \frac{\sqrt{var_k(x)}}{\mu_k} = \frac{(b-a)\sqrt{b_k}}{b + a_k(b-a)}$$

If $a = 0$ then $CV_k = \frac{\sqrt{b_k}}{(1+a_k)}$. Figure 3 shows that CV_k decreases in this case as well with the increase of k, the number of circuits a client chooses from. This results suggests that choosing the best-of-k circuit is a recommended practice, regardless of the skewness of the underlying distribution of the relay available capacities as a uniform distribution is flat compared to the exponential distribution.