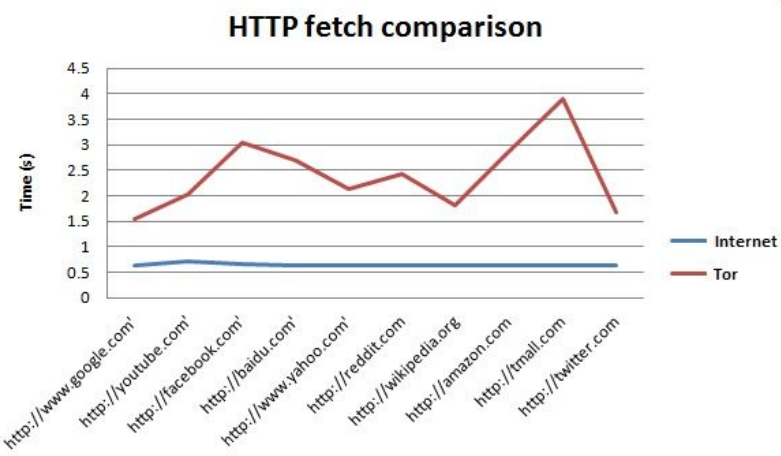
FASTor: A Low-Latency Tor Client

[Mid-Semester Report]

Muhammad Abdullah Malik (18100006)
Zain Imran (18100260)

Milestones achieved

- Read all the modern and older research papers related to our study and work.
- Set up programmable TOR controller environment (STEM)
- Setting up Shadow with TOR to virtualize a TOR network running on real-life parameters
- Creating custom paths on TOR with STEM library
- Retrieving the precise geographical location of each and every relay present on the TOR network and storing it into a hash table [9].
- Comparing the latency between an http fetch using simple internet and a TOR circuit for top 10 websites [8].
- Observing latency response time changes on different available circuits.



Challenges faced

The primary challenge that we faced was that with so much information and documentation at hand we were somewhat lost on how are we going to scale our problem statement and more importantly our solution. There were also issues in learning the STEM library.
We tackled the first issue by bringing our focus to only client side changes when it comes to improving Tor performance. Our primary focus of study and implementation has been based on the selection of relays and to re-implement LASTor which does relay selection based on geographic distances. We learnt the use of STEM library using the tutorial, but turned out to be more helpful was the APIs documentation.

The results of [3] indicate that the recently proposed LASTor relay selection strategy is unlikely to be effective on the live Tor network. It is likely that the evaluation method used by Akhoondi et al. [4] of making HTTP HEAD requests over LASTor paths did not realistically exercise the performance of those paths: a HEAD request is generally less than 1 KB, while typical web pages are two orders of magnitude larger. Since LASTor does not take the bandwidth of relays into account, its performance degrades drastically when a realistic traffic load is applied [7].

List of future tasks to be carried out (and a plan for the execution of the tasks)
We are halfway there re-implementing LASTor. With geographical distances of relays now at hand and the candidate paths we are now in the process of implementing a WSP algorithm that scores each of these candidate paths and then selects the best. We'll be skipping the AS-awareness from LASTor and will be focusing more upon other ways of selecting relays based on bandwidth. Our focus will also be towards selecting relays that have lesser load which links to efficient load balancing.
A lot of previous papers have explored improvement of TOR performance. However, each of them tackles a specific aspect of the system (e.g.path selection, client throttling, circuit scheduling, and flow/congestion control) hence, we propose combining some of these solutions to improve the performance of TOR. It is important to know that we are aiming to make these changes on the client side, with minimal or no change to the network. Among other advantages, this would also allow for easy deployment.

- Imposing strict rate limits on individual client connections at the network's ingress points but not limit connections originating from Tor relays. By carefully tuning these limits, interactive clients such as web browsers will see little effect, while bandwidth-intensive users (for example, file-sharers) will experience a significant decrease in throughput. This slowdown may provides incentives for file-sharers to operate their own relays through which they can bypass the network's rate limiting and in the process, increase the number of relays in the TOR network [1].
- Tor sequentially writes to sockets while ignoring the state of all sockets other than the one that is currently being written; and Tor writes as much as possible to each socket. By writing to sockets sequentially, Tor's circuit scheduler considers only a small subset of the circuits with writable data. This could lead to improper utilization of circuit priority mechanisms, which causes Tor to send lower priority data from one socket before higher priority data from another, thus indicating the ineffectiveness of circuit priority algorithm.
    - Implement Kernel-Informed Socket Transport (KIST) algorithm which can be used to relocate congestion from the kernel into Tor, where it can be properly managed [2].

References

[1] Moore, W. Brad, Chris Wacek, and Micah Sherr. "Exploring the Potential Benefits of Expanded Rate Limiting in Tor." *Proceedings of the 27th Annual Computer Security Applications Conference on - ACSAC '11* (2011): n. pag. Web.

[2] Jansen, Rob, et al. "Never Been KIST: Tor's Congestion Management Blossoms with Kernel-Informed Socket Transport." *USENIX Security*. Vol. 14. 2014.

[3] Wacek, Chris, et al. "An Empirical Evaluation of Relay Selection in Tor." *NDSS*. Vol. 13. 2013.

[4] Akhoondi, Masoud, Curtis Yu, and Harsha V. Madhyastha. "LASTor: A low-latency AS-aware Tor client." *Security and Privacy (SP), 2012 IEEE Symposium on*. IEEE, 2012.

[5] Research problem: measuring the safety of the Tor network: https://blog.torproject.org/blog/research-problem-measuring-safety-tor-network

[6] Dingledine, Roger, and Steven J. Murdoch. "Performance Improvements on Tor or, Why Tor is slow and what we're going to do about it." *Online: http://www. torproject. org/press/presskit/2009-03-11-performance. pdf* (2009).

[7] TOR ticket #6328 (Write a proposal for ASN based routing for clients): https://trac.torproject.org/projects/tor/ticket/6328

[8] Top 10 websites: https://www.quantcast.com/top-sites/US?userView=Public

[9] Geolocation from IP: http://dev.maxmind.com/geoip/legacy/geolite/