The Assessment of BitTorrent's Performance Using SDN in a Mesh Topology

Mandeep Kaur Guraya, Rupinder Singh Bajwa, Damian Vicino, Chung-Horng Lung
Department of Systems and Computer Engineering
Carleton University, Ottawa, Ontario, Canada
{mandeepguraya,rupinderbajwa}@cmail.carleton.ca, damian.vicino@inria.fr, chlung@sce.carleton.ca

Abstract – The basis of this paper is grounded upon the theoretical assumption, which states that the integration of BitTorrent with Software-defined Networks (SDNs) will degrade the performance of BitTorrent, instead of improving it. This was further evaluated by emulating this experiment on a Ring topology to verify the results. The experimental results, however, showed an unexpected behavior wherein BitTorrent outperformed under a SDN. This objective of this paper is to re-evaluate the behavior of BitTorrent in a Mesh topology instead. We have run 12 BitTorrent clients in an emulated SDN Mesh topology using Mininet and varied the virtual topology periodically by taking one link down at a time from the topology. The experimental results show when the logical topology changed periodically the performance is worse than when the topology is static for BitTorrent.

Keywords: Software-Defined Networks; BitTorrent; Mininet; Floodlight Controller

I. Introduction

File sharing and other types of content delivery could most reliably be achieved through peer-to-peer (P2P) applications, as they have become the most popular applications in the Internet. P2P communications could also be grouped in terms of how the content is delivered and consumed. BitTorrent is a content distribution protocol that enables efficient software distribution and peer-to-peer sharing of very large files using minimum Internet bandwidth, such as entire movies and TV shows, by enabling users to serve as network redistribution points.

A. BitTorrent

BitTorrent maximizes the transfer speed by gathering pieces of the file the peer wants, and downloading these pieces simultaneously from other peers who already have them. Torrent is a session established to transfer a single content of file to a group of peers [9]. Every such session of transfer is independent of each other. To establish a torrent, at least one seed is necessary in a swarm. Peers use swarming techniques to reproduce the file among each other. BitTorrent has generated great enthusiasm for P2P file distribution in academia and industry.

B. Software-Defined Networking

The network architecture called Software Defined Networking (SDN) is a separation of control plane and data plane from a physical entity. On the other hand, routers consist of both control and data planes together in the traditional networks. SDN has well-defined application programming interfaces (APIs) that enable SDN to handle multiple data elements in a single software program. The communication between the control plane and data plane is achieved through a protocol named OpenFlow [12].

This work is an extension to the research done with the motive of integrating SDN with the BitTorrent application to observe the impact SDN lays on BitTorrent's decision-making, and thus on its performance. The experiment conducted in this paper is, however, on a Mesh topology to verify the assumption made if BitTorrent shows a negative or positive impact on its performance with SDN's interaction.

II. Background

This section elaborates the theory that formed basis for our research work. The theory is based on an assumption that the implementation of SDN on the BitTorrent application would show a negative impact on BitTorrent's performance [13].

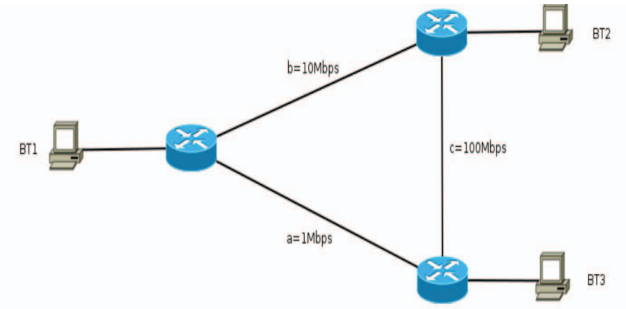

Fig. 1. An Illustrative Example [13]

The reason for this assumption is illustrated with the example as shown in Fig. 1. It demonstrates how a typical BitTorrent application works and when a SDN controller is implemented on top of it, the controller lays an impact on the

decisions taken by the BitTorrent. In this example, three BitTorrent clients are used and they are assumed to be in the same swarm. In other words, they are all interested in the same content to be shared. Also, BT1 is the seed, i.e., it has a complete copy of the file. Only two symmetric paths are maintained between these BitTorrent clients. Assuming that, once a piece is selected, client BT1 decides to unchoke BT3 (randomly chosen) without being aware of the underlying network. The limit for transfer between these two peers is kept to be 1Mbps. Soon after BT3 has been unchoked, it also unchokes BT2 to see if it can find a better option than the already ongoing flows. As BT2 gets unchoked, a new transfer starts using 10Mbps, and the client realizes it to be a better option than BT3 due to the higher bandwidth. Thus, it decides to choke BT3 back again. Reactive routing always assigns the oldest flow alive (that is, the first originated flow) the path with the highest bandwidth, thus the controller moves the flows so the first flow uses links b-c and the second flow uses a-c. After changing the flows, the connection between BT1-BT2 is limited to 1Mbps and the one between BT1-BT3 is limited to 10Mbps. However, the decision was taken to choke BT2. As a result, BitTorrent will keep using the lowest possible bandwidth between peers. If the Reactive Routing algorithm chooses again before the system stabilizes, BitTorrent will again choose the wrong one, and BitTorrent will never be able to obtain the max possible bandwidth.

The second problem arises when there are many switches in the path between a pair of nodes in a network. The change of paths created by the controller requires the flow tables to get updated, causes micro-cuts of service between the two nodes. This triggers the problem of anti- snubbing. Anti-snubbing is caused when a peer does not receive anything from a peer for more than a minute, it presumes to have been choked by it, thus choking it back in reciprocation. The triggering of anti-snubbing will effectively close the connection for a long time between 2 nodes, and hence, degrading the propagation rate.

These two problems suggested that a negative impact could be obtained by this interaction between SDN and BitTorrent. An experiment was conducted [13] by emulating a Ring topology to observe the BitTorrent's behavior to verify the assumption made. The results obtained were beyond expectation, as they showed an enhancement in the BitTorrent's performance. This, however, strikes a point that Ring topology would not be able to give a clear picture. For instance, the Ring topology was used in this experiment. Ring topology has only two directions to route its traffic from, which would either be clockwise or counter clockwise. Therefore, as a link goes down between two nodes, SDN does not have to calculate a path for the traffic to be routed from. It starts sending the traffic through the other direction. In practice, however, Mesh topologies are more common, where there could be many different paths to reach to the same destination. Therefore, as a link goes down, SDN has to do reactive routing according to the variations in the logical topology of the network. Thus, the outcome of the experiment reported in [13] became the motivation for this research work.

III. Experiment Setting

In order to evaluate the performance, there are certain evaluation criteria, which served as a base of this experiment. The main goal of BitTorrent is to have the fastest possible propagation of data among the peers and the maximum transfer rate. Hence, the first criterion, which is to be considered is the time taken to complete all the data transfer among the peers. Another important criterion to be considered is the number of downlink connections at any particular instant of time, which shows the maximum aggregated transfer rate among the peers in the BitTorrent.

Both these criteria need to be evaluated in order to have true measurements of BitTorrent's performance, as the changes in any one of them can affect the performance of BitTorrent.

A. Design Model

Having the criteria identified we designed an experiment model in order to measure these criteria. Two scenarios have been considered: Scenario 1 is to measure the performance without any failures. In other words, SDN controller does not have to perform any task to decide the forwarding path. Scenario 2 is to measure the performance when an arbitrary link is manually disconnected, and SDN has to re-calculate the new forwarding paths.

The design steps involved are as follows:

1. Create a Mesh topology with an SDN controller added on top of the network. The number of hosts used in the experiment is 12, where each host is connected to a switch.
2. Add BitTorrent clients to the hosts in the network.
3. Initialize a new seed in the network, consisting of a complete copy of the content to be transferred among the clients.
4. Start the transfer among the peers using the BitTorrent protocol, and the evaluation criteria discussed above was recorded.
5. Stop the experiment when all the peers have a copy of the content.
6. Repeat the above experiment, but disconnect one link in the topology (every 60 seconds), and let SDN controller decide the paths for the transfer among the peers.
7. Measure the performance of the experiment.

IV. Experimental Results

The comparison is made between the results collected when the topology never changes (using only BitTorrent protocols), and when the SDN controller is asked to change the topology every 60 seconds. Fig. 2 shows the total time for completed

transfers by BitTorrent client using the static topology and when the topology changes every 60 seconds.

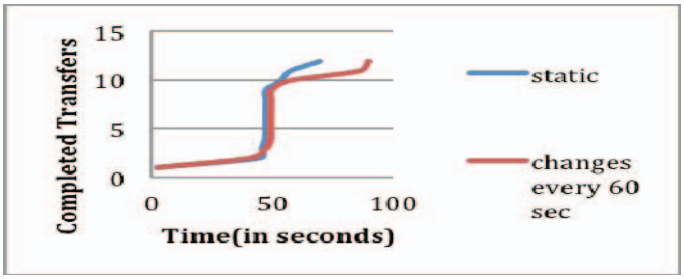

Fig. 2. Comparison of propagation of 10MB content through 12 nodes using different frequency of topology changes

The blue line shows the pattern when topology is never changed and it runs completely under the influence of BitTorrent. The red line shows the behaviour of the network when the controller is asked to change the topology every 60 seconds.

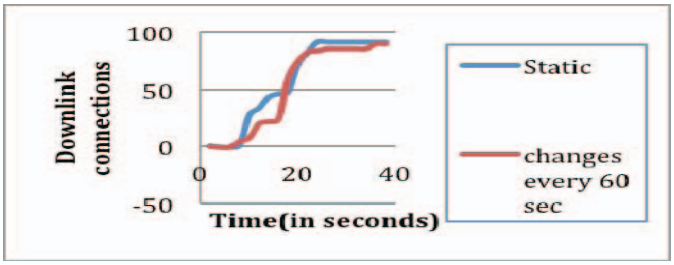

Fig. 3. Comparison of simultaneous open connections for download for distribution of 10MB content through 12 nodes using different frequency of topology changes

Fig. 3 shows the total number of downlink connections present at any time, first in the static network depicted by the blue line, and secondly, when the topology is changed every 60 seconds shown by the red line.

It is evident from the above figures that the BitTorrent is taking more time to transfer files when under the influence of SDN, thereby degrading its performance. A depicted in Fig. 2, it takes 69.156 seconds to transfer content among all the 12 hosts. Whereas, when the SDN is used, the topology is changed every 60 seconds, it takes 89.33 seconds for the same transfer. Thus, it can evidently be seen SDN is lowering down the BitTorrent's performance for the same amount of content to be transferred to the same number of peers. Also, at any point of time, the number of downlink connections available to any peer is higher when using the BitTorrent protocol alone in comparison to when used with SDN. The main reason is that the controller needs to take more time in re-calculating the routes in a Mesh topology after a link is disconnected.

V. Conclusions and Future Work

Theoretically, it was proven that the BitTorrent would show a negative impact in its performance under the influence of SDN. However, it was contradicted when the experiments were conducted on a Ring topology [13], in which the BitTorrent outperformed even under the impact of the SDN controller. In order to confirm this behavior in a more general case, these experiments were re-conducted on a Mesh topology in this paper, which supported the theory and showed a negative impact in the performance of BitTorrent when it is used together with the SDN architecture.

As a future research direction, one can re-conduct these experiments with a large number of clients in a lab environment. Another aspect to be considered would be to add background traffic to the network to make it more realistic to the real world. These could be considered as the future work of this experiment.

References

[1] P. Shah, JF Paris, "Peer-to-Peer Multimedia Streaming Using BitTorrent", *IEEE International Performance Computing and Communication Conf.,* April 2007, pp. 340–247.

[2] Y. Chu, S. Rao, H. Zhang, "A Case For End System Multicast", *IEEE Journal on Selected Areas in Communications,* 20(8), 2002, pp. 1456–1471 .

[3] BitTorrent*,www.en.wikipedia.org/wiki/Glossary_of_BitTorrent_term.*

[4] L.Guo, S.Chen, Z.Xiao, E. Tan, X. Ding, X. Zhang, "A Performance Study of BitTorrent-like Peer-to-Peer Systems", *IEEE Journal on Selected Areas in Communications,* 25(1), 2007, pp. 155–169.

[5] A. Legout, G. Urvoy-Keller, P. Michiardi, *Understanding BitTorrent: An Experimental Perspective*, Technical Report, INRIA, Nov. 2005.

[6] BitTorrent, *http://en.wikipedia.org/wiki/BitTorrent.*

[7] K.D. Vogeleer, D. Erman, A. Popescu, "Simulating BitTorrent", *Proc. of the 1st International Conf. on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops,* 2008, pp.1–7.

[8] B. Cohen, "Incentives Build Robustness in BitTorrent", *Proc. of the 1st Workshop on Economics of P2P Systems,* June 2003.

[9] Floodlight*, www.projectfloodlight.org/floodlight/*

[10] Myungki Shin, Kihyuk Nam, Hyoung-Jun Kim, "Software-Defined Networking (SDN): A Reference Architecture and Open APIs", *Proc .of International Conf. on ICT Convergence,* 2012, pp. 360–361.

[11] Open Network Foundation, "Software-Defined Networking: The New Norm for Networks", *ONF White Paper,* April 2012.

[12] E. Haleplidis, S. Denazis, O. koufopavlou, "Software-Defined Networking: Experimenting with the Control to Forwarding Plane Interface", *Proc. of European Workshop on Software Defined Networking,* 2012, pp. 91–96.

[13] D. Vicino, C.H. Lung, G. Wainer, O. Dalle, "Evaluating The Impact Of Software-Defined Networks' Reactive Routing on BitTorrent performance." *Procedia Computer Science,* vol. 34, 2014, pp. 668–673.