Cumhuriyet Science Journal
CSJ

e-ISSN: 2587-246X
ISSN: 2587-2680

Cumhuriyet Sci. J., Vol.40-3(2019) 762-767

# Comparison of Data Transfer Performance of BitTorrent Transmission Protocols

*Halil ARSLAN* [1,*], *Özkan CANAY* [2]

[1] *Cumhuriyet University, Engineering Faculty, Dept. of Computer Engineering, Sivas, TURKEY*
[2] *Sakarya University, Adapazarı Vocational School, Dept. of Computer Technologies, Sakarya, TURKEY*

Received: 19.09.2018; Accepted: 04.09.2019 *http://dx.doi.org/10.17776/csj.461655*

**Abstract.** BitTorrent, one of the distributed file sharing protocols, is regarded as one of the first examples of decentralized Internet philosophy and is among the important research areas in this context. TCP was initially used as the transport layer protocol in BitTorrent, and the transition to the uTP protocol was made because of the problems of latency and excessive bandwidth consumption. Later, with WebTorrent, which is a BitTorrent protocol adapted to the web, WebRTC was proposed as a transport layer protocol. Thus, BitTorrent protocol is enabled to work directly through Internet browsers without using any plugin. In this study, the data exchange sizes in the torrent shares of these three transmission protocols have been compared and the advantages and disadvantages of these protocols were demonstrated in this context.

***Keywords***: BitTorrent, WebTorrent, TCP, uTP, WebRTC.

## BitTorrent İletim Protokollerinin Veri Aktarım Başarımlarının Karşılaştırılması

**Özet.** Dağıtık dosya paylaşım protokollerinden BitTorrent, merkezi olmayan internet felsefesinin ilk örneklerinden olarak kabul edilmekte ve bu bağlamda önemli araştırma alanları arasında yer almaktadır. BitTorrent'te iletim katmanı protokolü olarak başlangıçta TCP kullanılmış, gecikme ve bant genişliğinin aşırı tüketilmesi problemlerinden dolayı uTP protokolüne geçiş yapılmıştır. Daha sonra BitTorrent protokolünün Web'e uyarlanmış hali olan WebTorrent ile iletim katmanı protokolü olarak WebRTC önerilmiştir. Bu sayede herhangi bir eklenti kullanmadan doğrudan internet tarayıcıları üzerinden BitTorrent protokolünün çalışması sağlanmıştır. Yapılan çalışmada, bu üç iletim protokolünün torrent paylaşımlarındaki veri alışveriş boyutları karşılaştırılmış ve bu bağlamda avantaj ve dezavantajları ortaya konulmuştur.

***Anahtar Kelimeler***: BitTorent, WebTorrent, TCP, uTP, WebRTC.

## 1. INTRODUCTION

The BitTorrent protocol supports file sharing over peer-to-peer networks (P2P) using the TCP [1] and uTP [2] protocols. On the web side, the same function is performed with WebRTC via WebTorrent [3]. When the TCP protocol is used as the transmission protocol, BitTorrent protocol clogs the bandwidth in DSL and cable modems, and causing delays on interactive communications [4]. For these reasons, the problem of consuming the full bandwidth is solved by using uTP as the transmission protocol [4]. In the following years Aboukhadijeh [5] suggested WebTorrent by making some protocol changes to run the BitTorrent protocol over WebRTC. WebTorrent is a simple JavaScript API that runs in the browser without the need for an extra plugin [6].

* Corresponding author. *Email address: harslan@cumhuriyet.edu.tr*

WebTorrent uses RTCDataChannel of WebRTC for P2P communication of BitTorrent protocol. Thus, WebRTC has been used as a data transmission layer for the BitTorrent protocol and the WebTorrent protocol has emerged [3]. In this way, man in the middle attacks are prevented [7] [8] and completely browser based torrent clients have emerged. This study examines the transport layer protocols recommended for the BitTorrent protocol, and compares the packet numbers and bandwidth usage that affect network performance.

## 2. DATA TRANSMISSION ON BITTORRENT

### 2.1. TCP

BitTorrent was designed by Bram Cohen in 2001 to seamlessly integrate with the web by defining the content URL for file sharing. When TCP (Transmission Control Protocol) is used as the transmission layer in the BitTorrent protocol, multiple TCP connections are opened and shared [9]. The most important advantage over HTTP is that it supports a large number of downloads from different sources with a slight increase in the file resource load. This causes delays in the Internet connection due to the filling of the sender buffer in DSL and cable modems [10].

Although this problem is solved by constantly limiting BitTorrent bandwidth, it is necessary for users to rearrange the restriction when they pass to different bandwidth networks [1]. At the same time, when bandwidth utilization increases, fixed limitations cause repeat latency and interruption. If the bandwidth utilization drops, the unused bandwidth will not be evaluated. Due to these problems, uTP protocol has been developed to dynamically adjust the bandwidth usage [11].

### 2.2. uTP

TCP protocol includes Additive Increase/ Multiplicative Decrease (AIMD) [12] to use equal bandwidth between TCP connections. However, BitTorrent uses a lot more bandwidth than applications that use a single TCP connection because it builds multiple TCP connections to peer nodes with the P2P architecture. Basically, to solve this problem BitTorrent has developed a UDP-based micro transport protocol (uTP) [13] based on new congestion control (Low Extra Delay Background Transport-LEDBAT) [14].

The uTP protocol dynamically runs on the unused portion of the bandwidth. The protocol controls the connection flows with the sliding window algorithm and preserves the data integrity with consecutive sequence numbers [15]. uTP basically uses a one-way delay measurement as the congestion control gain. The sender places a 32-bit time stamp in the data packet defining the sending time. The receiver calculates the one-way delay measurement and stores it as a delay vector. If the smallest value in the stored delay vectors is less than 100, the window size is increased. If it is higher than 100, it is decreased. This congestion control approach does not fill the send buffer and does not cause interruptions and delays in the user's Internet connection [11].

### 2.3. WebRTC

WebRTC (Web Real-Time Communications) is a JavaScript-based, open source free project that provides real-time communication without the need for additional extensions on browsers [16]. It is supported by browsers such as WebRTC, Firefox, Opera and Chrome, and platforms like Android and IOS, which are rapidly spreading at the point of browser support with HTML5 [17]. WebRTC provides a set of features for peer-to-peer communication and video conferencing in HTML, for instance connecting to remote peers using NAT traversal technologies such as ICE, STUN, and TURN, and sending data directly to remote users or receiving fragments from remote peers [18].

The main purpose of WebRTC is to access peripherals such as cameras and microphones in a secure way through the web application and to provide real-time data flow between peers. WebTorrent, which uses WebRTC as the transmission protocol, is a browser-based special torrent client that uses WebRTC data channels to connect to the BitTorrent network and does not need any additional downloads. In this sense, WebTorrent can offer direct torrent downloads from browser to browser using WebRTC peer connections and WebSocket-based tracker protocol with partial changes to the BitTorrent protocol. However, WebRTC based WebTorrent and TCP / UDP based BitTorrent clients cannot communicate directly with each other. Transmission applications called hybrid clients are needed for this [19].

## 3. METHODOLOGY AND MODEL

In order to compare the packet numbers and sizes that WebRTC, uTP and TCP protocols used during the transfer in BitTorrent usage, an appropriate model has been prepared to the network topology shown in Figure 1.

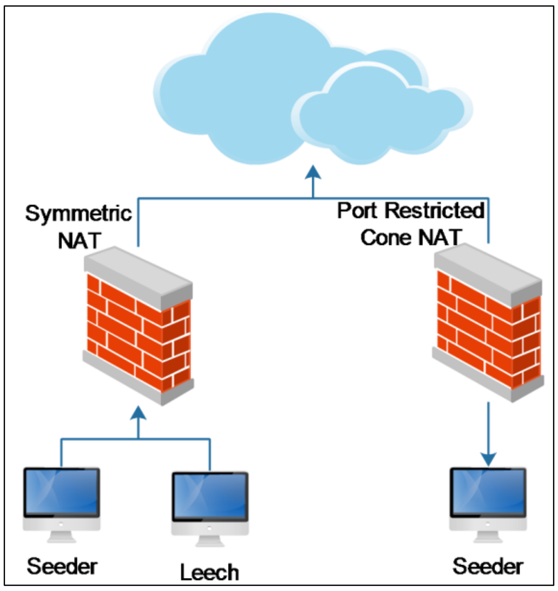

**Figure 1.** Application model

BitTorrent version 7.9.9 was used as the official processor of the BitTorrent protocol for TCP communication. For uTP communication, BitTorrent client is configured and transfer is made via uTP. The JavaScript-based WebTorrent client for WebRTC has been used over Google Chrome version 55.0.2883.87. For TCP and uTP transmissions with the BitTorrent client, all incoming and outgoing TCP and UDP connections have been allowed on the firewall of client's operating system. No settings have been made in the browser or operating system for WebRTC transmission.

The BitTorrent client has been used with default settings for uTP forwarding; by default, the security settings will support unencrypted connections on outbound connections, and encrypted and unencrypted connections on incoming connections. In order for BitTorrent client to be able to transmit TCP, the default transmission method uTP protocol is set to passive and the TCP protocol is activated. By default, security settings are configured to allow unencrypted connections for outgoing connections and to support encrypted and unencrypted connections for incoming connections. End-to-end security on the WebRTC is not optional, as it is standard, no extra security settings have been set.

The first sharer is located under the Port Restricted Cone NAT, and the internet connection is realized on this way. The second sharer and client are located under Symmetrical NAT. Two sharers and one client are used for each transmission protocol. Two files of 100KB and 1MB in size, have been uploaded to both sharers and downloaded by the client respectively. Two files were requested separately for TCP, uTP and WebRTC protocols and network data were monitored. Six downloads were made in total. Packets transferred from starting client side communication until completion of downloading are captured, packet numbers and packet sizes are calculated and compared for each protocol separately.

## 4. RESULTS AND DISCUSSION

In this study, the use of WebRTC, TCP and uTP as the transmission layer in the BitTorrent protocol is discussed. This transport layer protocol has been examined through the methodology and model presented in Chapter 3. The first sharer made a 100KB file share with BitTorrent uTP, and the second sharer shared this downloaded torrent via itself. After this process, two shareholders of the torrent with the same content were created. The client started downloading torrent content using uTP. The client receives part of the file from the shareholder on its own network and the other part from the shareholder behind a different NAT. Communication between the nodes behind Symmetric NAT and Port Restricted Cone NAT has been successfully established. Subsequently, 1 MB file sharing was performed over the same model and the same transmission protocol.

The BitTorrent client for TCP transmission is configured to transmit TCP only, 100Kb and 1Mb files are downloaded separately from the two sharers. NAT transitions have been successful. As a result of this communication, the bandwidth and number of packets used by the client are shown in Table 1.

**Table 1.** Bandwidth and number of packets on BitTorrent TCP connection

| File | Bandwidth | Num. of Packets |
|---|---|---|
| **100KB** | 112,482 byte | 150 |
| **1MB** | 1,119,539 byte | 1,140 |

BitTorrent client for uTP transmission is configured, 100KB and 1MB files are downloaded separately from two sharers. Bandwidth and number of packets used by the client as a result of this communication are shown in Table 2.

**Table 2.** Bandwidth and number of packets on Bittorrent uTP connection

| File | Bandwidth | Num. of Packets |
|---|---|---|
| **100KB** | 119,772 byte | 197 |
| **1MB** | 1,120,156 byte | 1,010 |

For WebRTC transmission, a browser-based application that transmits WebTorrent via WebRTC runs on two shareholders; 100KB and 1MB torrents were uploaded to the first sharer. The second sharer has downloaded the torrent with the created torrent link via WebRTC and started sharing. As a result of the process, two shareholders of the file were created. Some parts of the file are downloaded from the sharer on the same network and the other parts are downloaded from the sharer behind the different NAT. The NAT transition was successful.

Since the client network is Symmetrical NAT, and the other sharing network is Port Restricted Cone NAT, it has been observed that the TURN server has been activated and data is relayed via this server. For this reason, it has been observed that STUN and TURN processes have been successfully performed in accordance with the ICE framework. The bandwidth and number of packets used by the client as a result of the transmission through WebRTC via WebTorrent are shown in Table 3.

**Table 3.** Bandwidth and number of packets on WebRTC Transmission

| File | Bandwidth | Num. of Packets |
|---|---|---|
| **100KB** | 136,246 byte | 345 |
| **1MB** | 1,680,905 byte | 5,467 |

After the completion of the transfer within using all the protocols, bandwidth and number of packets used for the 100KB file are shown in Figure 2 and Figure 3, respectively.

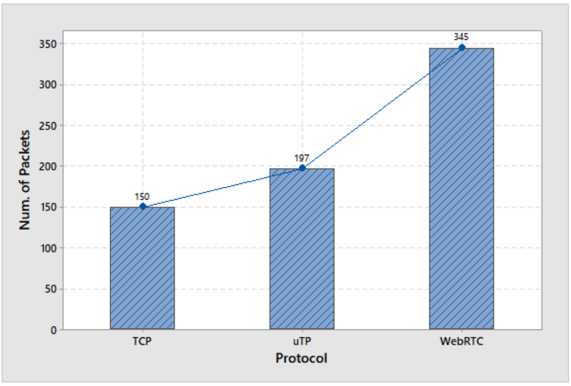

**Figure 2.** Total number of packets for 100KB file transfer

In the 100KB file transfer, TCP has completed the transfer with 150 package, uTP used 197 packet, and WebRTC used 345 packets for this communication.

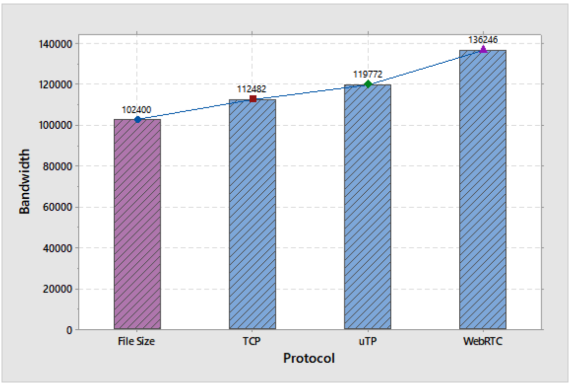

**Figure 3.** Bandwidth utilization for 100KB file transfer

In this sharing process, the total size of data sent outside the file size (overhead) was approximately 10KB (10,082 byte) in the TCP protocol, 17KB (17,372 byte) in the uTP protocol, and 33KB (33,846 byte) in the WebRTC protocol.
After all the transfers are complete with all protocols, bandwidth utilization and number of packets for the 1MB file are presented comparatively in Figure 4 and Figure 5.

uTP protocol works on UDP basis and it is observed that the added control packets and the congestion control algorithm increase the communication size because the transferred data should be sent lossless to the other side and thus use more packets and bandwidth than the TCP protocol. The WebRTC protocol seems to use quite a lot of packets and bandwidth compared to uTP and TCP. It can be said that the main factor that increases the number of packets is the signaling packets and the data fragments sent with low payload caused by the activation of the TURN Server.

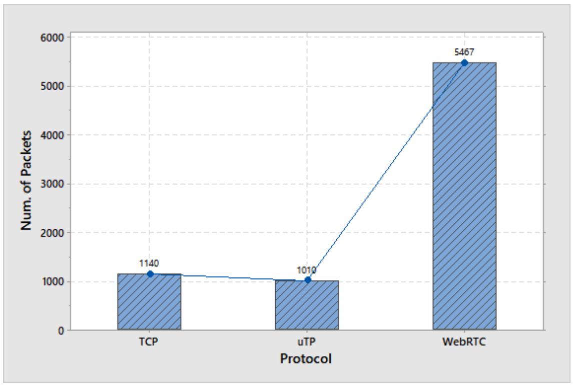

**Figure 4.** Total number of packets for 1MB file transfer

It is considered that the reason for the excessive use of bandwidth in WebRTC is the dTLSv1.2 protocol used for end-to-end security after signaling and the overhead caused by the TURN server.

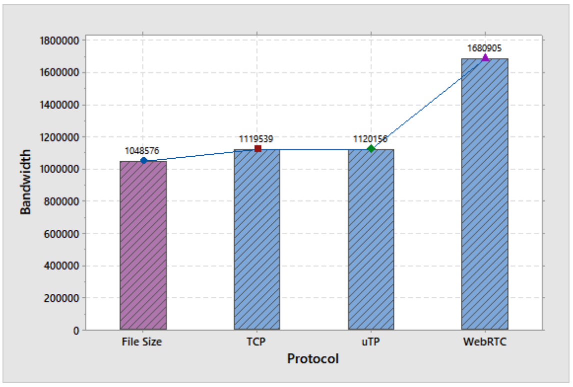

**Figure 5.** Bandwidth utilization for 1MB file transfer

## 5. CONCLUSION

Although it is observed that WebRTC is the protocol that has the most packet traffic and bandwidth usage in file transfers made with BitTorrent protocols the use of this protocol seems to be advantageous compared to the uTP and TCP protocols in the use of BitTorrent because it provides significant convenience at the end-user point, such as platform, installation and configuration requirements. Over time, it is expected that WebRTC's connection efficiency will be improved. Also, it's observed that on the BitTorrent Protocol running on TCP and uTP, encrypted connection for connection security is optional and can be running by giving special permissions to the operating system firewall.

BitTorrent (WebTorrent) protocol running on WebRTC brings significant advantages such as working on web browsers, no need for additional software installation, no need for specific firewall permission, providing dTLSv1.2 protocol as standard for connection security, user security, and end-to-end confidentiality. In future studies, the security parameters of BitTorrent transmission protocols can be examined and their advantages and disadvantages can be demonstrated.

## ACKNOWLEDGMENTS

This study is the result of the studies carried out at the Detaysoft R&D Center. We would like to thank Detaysoft for its support.

## REFERENCES

[1] Gong, Y., Rossi, D., Testa, C., Valenti, S., & Täht, M. D. Fighting the bufferbloat: on the coexistence of AQM and low priority congestion control. Computer Networks, 65 (2014) 255-267.

[2] Adamsky, F., Khayam, S. A., Jäger, R., & Rajarajan, M. Stealing bandwidth from BitTorrent seeders. Computers & Security, 46 (2014) 126-140.

[3] Aboukhadijeh, Feross. Streaming torrent client for the web. https://github.com/feross/webtorrent. Retrieved February 1, 2017.

[4] Norberg, A. uTorrent transport protocol. http://www.bittorrent.org/beps/bep_0029.html. Retrieved January 30, 2017.

[5] Aboukhadijeh, F. WebTorrent: Using WebRTC and Mad Science to Bring BitTorrent to the Web. Software Delivery Craftsmanship Matters (2014)

[6] Burgstaller, F., Derler, A., Kern, S., Schanner, G., & Reiter, A. Anonymous communication in the browser via onion-routing. IEEE 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC) (2015) 260-267.

[7] A Study of WebRTC Security. NTT Communications project. http://webrtc-security.github.io/. Retrieved January 29, 2017.

[8] Feher, B., Sidi, L., Shabtai, A., & Puzis, R. The Security of WebRTC. arXiv preprint: 1601.00184. (2016) 1-10.

[9] Cohen, B. Incentives build robustness in BitTorrent. Workshop on Economics of Peer-to-Peer systems, 6 (2003) 68-72.

[10] Testa, C., & Rossi, D. Delay-based congestion control: Flow vs. BitTorrent swarm perspectives. Computer Networks, 60 (2014) 115-128.

[11] Adamsky, F. Analysis of bandwidth attacks in a bittorrent swarm (Doctoral dissertation). University of London. (2016)

[12] Chiu, D. M., & Jain, R. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. Computer Networks and ISDN systems, 17(1) (1989) 1-14.

[13] Rossi, D., Testa, C., Valenti, S., & Muscariello, L. LEDBAT: the new BitTorrent congestion control protocol. Proceedings of IEEE 19th International Conference on Computer Communications and Networks (2010) 1-6

[14] Testa, C., & Rossi, D. On the impact of uTP on BitTorrent completion time. IEEE International Conference on Peer-to-Peer Computing (2011) 314-317.

[15] Adamsky, F., Khayam, S. A., Jäger, R., & Rajarajan, M. P2P file-sharing in hell: exploiting BitTorrent vulnerabilities to launch distributed reflective DoS attacks. 9th USENIX Workshop on Offensive Technologies (2015)

[16] Arslan, H., Tuncel, S., & Yuksek, A. G. Comparison of the Web based multimedia protocols for NAT traversal performance. IEEE 23nd Signal Processing and Communications Applications Conference (SIU) (2015) 915-918.

[17] WebRTC - Real Time Communications in the browser. http://www.webrtc.org. Retrieved February 1, 2017.

[18] Hickson, I. WebRTC 1.0: Real-time Communication Between Browsers. http://www.w3.org/TR/webrtc/. Retrieved February 1, 2017.

[19] Aboukhadijeh, F. WebTorrent. JSConf Asia. https://2014.jsconf.asia/. Retrieved February 1, 2017.