An Experimental Evaluation of Data Distribution Applications in Tactical Networks

Enrico Casini[1], Giacomo Benincasa[1], Alessandro Morelli[2], Niranjan Suri[1,3], Maggie Breedy[1]
[1]Florida Institute for Human & Machine Cognition (IHMC), Pensacola, FL USA
[2]University of Ferrara, Ferrara, Italy
[3]U.S. Army Research Laboratory (ARL), Adelphi, MD USA
{ecasini, gbenincasa, nsuri, mbreedy}@ihmc.us
{alessandro.morelli}@unife.it
{niranjan.suri.civ}@mail.mil

***Abstract*—Tactical Edge Networks are one of the most challenging communication environments, where node mobility, constrained devices, and the wide use of wireless ad hoc channels for communications cause frequent link disruption and network partitioning. Additionally, applications running in Tactical Edge Networks often follow one-to-many and many-to-many communication models that require the nodes' cooperation to enable data delivery. Many studies proposed the use of peer-to-peer systems in mobile ad hoc networks to remove any dependency on centralized nodes and to cope with link disruptions and network partitioning phenomena that derive from the extreme dynamicity of Tactical Edge Networks. In this paper, we compare the performance of file sharing in mobile ad hoc networks using an implementation of the BitTorrent peer-to-peer protocol and DisService, a middleware for information dissemination specifically designed for Tactical Edge Networks. DisService supports applications with routing and peer-to-peer content delivery in mobile ad hoc networks. We ran several experiments in an emulated environment to evaluate the performance of each solution when sharing large files. The obtained results show that TEN-specific solutions such as DisService can outperform traditional peer-to-peer solutions like BitTorrent.**

***Index Terms* —Tactical networks, Mobile ad hoc networks, Peer-to-peer computing, Wireless application protocols**

I. Introduction

Tactical Edge Networks (TENs) represent an extremely challenging networking environment. They are composed of a variety of networks, such as Wireless Sensor Networks (WSNs) and Mobile Ad hoc Networks (MANETs), which provide a highly heterogeneous communication environment. Additionally, frequent node mobility, the presence of constrained devices, and unstable wireless channels, which provide low bandwidth communications with variable latency, cause links in TENs to break easily.

The heterogeneity of TENs also includes the types of applications and their traffic. Communications performed by tactical applications range from small soldier-to-soldier chat messages, to the distribution of Command-and-control messages from the Operations Center (OC) to a large subset of the nodes in the TEN and the collection of large volumes of sensor data from the edge back to the OC. In such a scenario, because a path between source and destinations may not always exists, nodes need to cooperate to share their communications and storage resource to increase the availability of the information on the network, and therefore increase the delivery performances.

Several studies propose the use of peer-to-peer (P2P) systems to provide file sharing functionalities in MANETs [7] [8] [9]. P2P applications significantly reduce, or remove altogether, their dependency on centralized nodes. Additionally, P2P systems naturally replicate information in the network, thus increasing its availability. These characteristics make P2P systems fit TENs particularly well, to counter the common phenomena of network partitioning and link disruption.

Nonetheless, there are also some important differences between P2P systems and MANETs. In fact, P2P applications normally rely on an overlay network of peers participating in the file sharing process, whose setup and maintenance introduce a significant overhead in the network. Moreover, most P2P solutions make use of TCP for transferring file chunks between nodes of the overlay network. However, TCP suffers from poor performance in MANETs [10] and causes connections to break down as soon as any endpoint becomes unreachable, even if temporarily [20], which is a common event in TENs.

In this study, we compare the performance of Ttorrent [14], an open source P2P file sharing solution based on the BitTorrent protocol, with the Agile Computing Middleware (ACM) DisService [4]. BitTorrent is a relevant candidate to this evaluation as it has proven extremely successful in infrastructure networks [2], and its adoption in TENs has been proposed in a significant number of publications [6] [12] [16]. On the other hand, DisService is an implementation of an information dissemination middleware specifically designed for extremely dynamic communication environments, such as TENs. DisService provides applications with efficient solutions to both routing and P2P content delivery in MANETs that we devised to take advantage of the similarities between the two systems. Because DisService was designed for highly mobile network, by default it does not construct an overlay network, as the maintenance costs would be likely to exceed the benefits.

We performed a set of experiments in an emulated networking environment to compare the performance of file

sharing using Ttorrent and DisService. Our results show that TEN-specific solutions such as DisService can perform significantly better than traditional P2P applications.

II. P2P Systems and MANETs: a Comparison

Although P2P systems and MANETs target different problems at different abstraction levels (application vs network), they both provide distributed and self-organizing solutions. Following this principle, P2P networks and MANETs share several important design aspects.

First, nodes are peers. In P2P file sharing systems, peers that joined the P2P overlay network can issue requests for file download or serve other peers' requests. Those tasks are not mutually exclusive, and it happens often that peers download one or more files while uploading file segments to other peers in the network. In MANETs, nodes collaborate to provide physical network connectivity and route packets towards the destination and back to the source. Additionally, any node in the network can take up the role of source or destination in a communication, potentially at the same time.

Another common characteristic concerns the dynamicity of the network and the frequent changes in network topology. Even in infrastructure environments such as the Internet, the topology of P2P overlay networks changes continuously because of peers joining and leaving the network. The authors of [1] state that, from their measurements, the median session duration of peers in a P2P system is 60 minutes. Similarly, also in MANETs peers frequently join and leave the network. The main reason is mobility, which might cause disconnections when peers fall out of the transmission range of their neighbors. Another reason is the limited energy available: most mobile devices are battery-operated, thereby they could run out of energy or voluntarily switch off their network interface cards in order to extend the remaining operation time. A third reason would be to maintain radio silence to reduce the possibility of detection.

Finally, the usage, to some extent, of multicast/broadcast communications and/or flooding is another common trait of both systems. Many P2P solutions use broadcast to update their knowledge of the overlay network, start a file search, or discover local peers (as in the Local Peer Discovery feature of the BitTorrent protocol). Likewise, nodes in MANETs typically make wide use of broadcasting and flooding in order to discover new resources, update their routing table, advertise neighbors about their presence, or disseminate data.

Despite these commonalities, P2P systems and MANETs still maintain some important differences. First, while P2P systems operate at the application level and build a virtual (overlay) network that peers need to join in order to share resources, MANETs are a network level concept that consists of nodes that collaborate to provide multi-hop connectivity capabilities to all members of the MANET. Therefore, P2P applications require support from a physical network infrastructure in order to build the overlay network for file sharing. In the Internet, TCP and UDP over IP can provide all the necessary communication abstractions to P2P applications and peers can rely on a stable, reliable wired network infrastructure. In a wireless mobile environment, this support is usually provided by routing protocols designed for MANETs. However, as previously described, MANETs in TENs do not provide the stability that is enjoyed by P2P systems operating in the Internet.

III. The BitTorrent Protocol

BitTorrent [2] is a network communications protocol that enables peer-to-peer file sharing and is exceptionally popular in the context of distributing data over the Internet. BitTorrent is also one of the most used protocols for transferring large files according to [3].

BitTorrent allows peers to join a "swarm" of hosts to upload and download from each other simultaneously. This enables faster download speeds and, perhaps most notably, decreases the impact of distributing large files, in opposition to the standard approach of downloading a file from a single origin server. By utilizing this distributed approach, multiple geographically distributed personal computers replace large-scale and purpose built servers and data centers in the efficient distribution of files to many recipients with lower bandwidth usage than single source, multiple mirror approach. This also prevents large concentrated spikes in network traffic, keeping resource usage lower for all peers.

A user who wants to share some content with other users first creates a torrent descriptor file (.torrent) associated to that content and then distributes it through some out-of-band mechanism, for instance by making it available publicly on the web or delivering it via e-mail to other users. At this point, the user can load the torrent descriptor file on the BitTorrent client, which in turn contacts a "tracker" specified at the beginning of the file. The tracker is a special server that keeps track of the connected nodes of the P2P network. The tracker shares their IP addresses with other BitTorrent clients in the swarm, allowing them to connect to each other. Subsequently, the user who has the content will make the relative file(s) available through a BitTorrent client node that will act as a seed. Users that retrieved or received the torrent descriptor file can load it into their BitTorrent client, which, acting as a peer or "leecher", can start the download by connecting to the seeds and/or other peers.

In BitTorrent, files are divided into smaller chunks (in the order of a thousand per file), and the downloaders of a file exchange its chunks by uploading and downloading them in a tit-for-tat-like manner to prevent parasitic behavior. Although peers favor this behavior, it is not mandatory, as it could prevent new peers to ever start downloading the file. Furthermore, each peer in the system is responsible for maximizing its own download rate by contacting suitable peers, and there is a high probability that peers with high upload rates will also be able to download with high speeds. Once a peer has finished downloading a file, it may decide to become a pure seeder for as long as desired, to increase content availability in the network and help other peers accomplish their goal as well.

IV. ACM DISSERVICE

The Dissemination Service (DisService) is part of the Agile Computing Middleware (ACM) [20];it provides applications with a P2P, publish-subscribe, message-oriented, store-carry-and-forward, dissemination service that was designed for point-to-multipoint communications in challenged wireless networks [4]. In order to support the transmission requirements for different data types, DisService supports all the combinations of reliable/unreliable, and sequenced/non-sequenced communications. The decision on which communication pattern to utilize is made individually by the recipients of the communication on a per subscription basis: while some of the recipients of the messages published in a certain group may require reliable delivery, others may not. For instance, dismounted soldiers equipped with battery-powered communication radios may be interested in receiving only the latest red-force tracking information, whereas intelligence units that are located at the headquarters may be interested in receiving all of the red-track information in order to better analyze the enemy's moves.

Because of these reasons, DisService adopts reliable-reception as opposed to TCP-like reliable transmission, by employing selective negative acknowledgement (SNACK). Reliable reception allows for seamless handling of reliable communications: a sender's only concerns are sending data and re-transmitting the specified parts whenever requested. If a recipient node moves away and misses some messages, it will request them once it is back in communication range. Furthermore, reliable reception allows recipients to request missing parts from other nodes as opposed to only the original publisher. This latter feature is particularly beneficial, especially in combination with aggressive caching policies.

DisService implements opportunistic listening [5], which enables nodes to listen to any message transmission performed by other nodes and store overheard messages for later usage by the node itself, or others. As a consequence of DisService's design focus on wireless networks, we made use of local broadcast messages given that the costs of unicast and local broadcast transmissions are equivalent; this further increases the availability of information, and makes the system more resilient to partitioning and disconnections. Finally, the use of self-contained and self-describing messages, in combination with opportunistic-listening, aggressive caching, and reliable reception, creates an effective P2P data distribution service.

V. EXPERIMENTAL EVALUATION

This Chapter presents the experimental scenario we used for our tests and the results we obtained. It is especially important to note that we designed our experiment to evaluate and compare the performance of the *data transfer* phase with Ttorrent and DisService. Therefore, we decided to keep the scenario very simple, to reduce as much as possible the effects of other factors, such as node mobility, the routing protocol chosen, the number of nodes involved, and so on.

For the purpose of this paper, it is worth mentioning that our original experimental evaluation plan included CodeTorrent [11] other than Ttorrent to provide a fairer comparison with DisService. CodeTorrent was in fact designed for mobile peer-to-peer systems and it is optimized for vehicular ad hoc networks. We experimented with the open-source version of CodeTorrent in our testbed, but we encountered several problems during the transfer of simple files. We established that, in all likelihood, the code was not designed for a real network environment since it did not deliver packets properly. We look forward to experiment with any upcoming version, and add other protocols to our experimental scenario.

A. *Experimental Scenario*

Our experimental scenario consists of a testbed of 20 nodes fully interconnected by emulated tactical links. Among these 20 nodes, a single sender transmits a single large file to the remaining 19 nodes. This scenario represents a typical dissemination pattern where one node has some new information (e.g., an Operations Center, or a sensor) that needs to be delivered to all the other nodes.

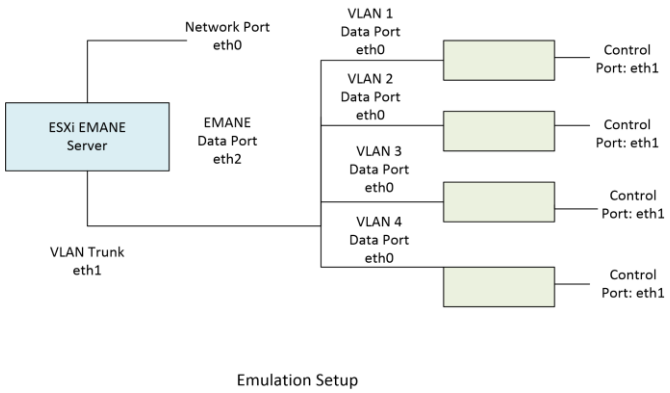

Figure 1 - The Emulation Setup

Differently from previous studies [6] [8] [16] [17], the focus of our evaluation is on the performance of the data transfer protocols only; therefore, we purposely configured the network in a static, fully connected mesh, so that no routing protocols are required in order to provide multi-hop connectivity among the nodes. In fact, protocols like OLSR [18] or AODV [19], which are needed in a real-world scenario, would introduce overhead into the network and increase the communication delay to build and maintain the routing tables on each node. Since DisService does not rely on any underlying routing protocol, as it takes care of routing by disseminating messages in the network, adding an additional layer only for BitTorrent would affect the fairness of our experiment and we would not be able to narrow down our comparison to the performance of the data transfer phase of the two solutions.

We used the Extendable Mobile Ad-hoc Network Emulator (EMANE) [13] to control the network connectivity. For the experiment, we relied on the EMANE Comm Effect Model to change the packet loss values between the links. The packet loss values, expressed in terms of packet error rate, were 0%, 5%, 10%, and 15%, equivalent to 100%, 95%, 90% and 85% reliability, respectively.

The experiment consisted of four physical machines running one virtual server and four virtual machines. The emulation setup for one of the four servers is shown on Figure 1. Each physical machine has a VLAN setup to run one EMANE server and the four test VMs running the DisService and Ttorrent code. The server runs the emulation and is responsible for generating the Comm Effects for each packet loss value. We developed customized test programs using the DisService and Ttorrent Java libraries. Both versions of the test program send three videos files of different lengths (50, 200, 500 MB) over the emulated network for the four values of packet loss.

During the experimental phase, we ran approximately 30 sessions for all the combinations of file size and packet loss. In order to have a fair benchmark of the experiment, we measured the throughput, defined as the rate of messages successfully delivered over the communication channel, and the amount of bandwidth consumed by DisService and Ttorrent to accomplish the file transfer. As a further requirement, we set a maximum time frame of 240 minutes for the file transfer to finish. If the transfer would not complete within the given 240 minutes, we would proceed with the next run and mark the record for that specific combination of file size and packet loss as not completed. We find this choice for the time frame to be suitable in order to have a fair comparison in TEN scenarios.

In our scenario, 19 out of 20 nodes act as receivers, hence we expect the data being sent only once by DisService, but 19 times via Ttorrent (we are not considering packet loss in our estimate). In fact, DisService uses a mechanism similar to many-cast [15] to send the data from the sender to the 19 receivers, whereas applications based on BitTorrent rely on point-to-point communications. This makes DisService likely to be in the order of 19 times more efficient than Ttorrent in terms of total bandwidth consumption.

Finally, it is relevant to point out that the standard version of BitTorrent needs a tracker to function properly [9], which is to be considered as a single point of failure in the P2P network. Although this would be critical for several other scenarios and applications, given the static network topology we used in our tests, for the purpose of this experiment we will not acknowledge this limitation as a compromising factor.

B. *Experimental Results*

Figures 2, 3, and 4 show the throughput achieved by Ttorrent and DisService in our tests when varying the link reliability. The results exhibit the advantage of using DisService over the BitTorrent protocol in this setup, which grows as the rate of packet loss increases over the tactical links emulated by EMANE. It is worth noticing that, for the combination of 500 MB and 15% pack loss, we do not have results for Ttorrent since the test program did not complete the file transfer within the time limit. Furthermore, this shows how COTS P2P solutions such as Ttorrent and, more in general, the BitTorrent protocol, might not be a good fit for tactical scenarios, where extremely high file transfer latencies are unacceptable for the purpose of the mission's goals.

In terms of bandwidth consumption, DisService shows another clear advantage over Ttorrent. By exploiting local broadcast or many-cast and adopting several different strategies to minimize its bandwidth usage, like opportunistic listening, caching, and consolidating retransmissions for lost packets [4] [5], DisService shows very low usage of bandwidth, especially when compared to the standard BitTorrent protocol adopted in this scenario.

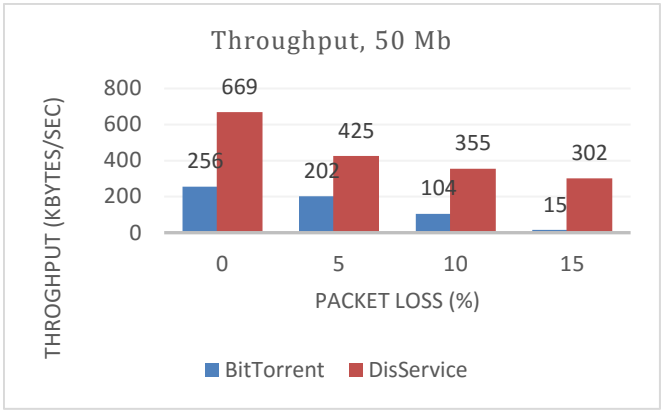

Figure 2 - Throughput measured transferring a 50 MB file

Figures 5, 6, and 7 show how DisService outperforms Ttorrent, by as much as 20 times, even at the lowest level of packet loss. The benefits of using solutions based on many-cast,

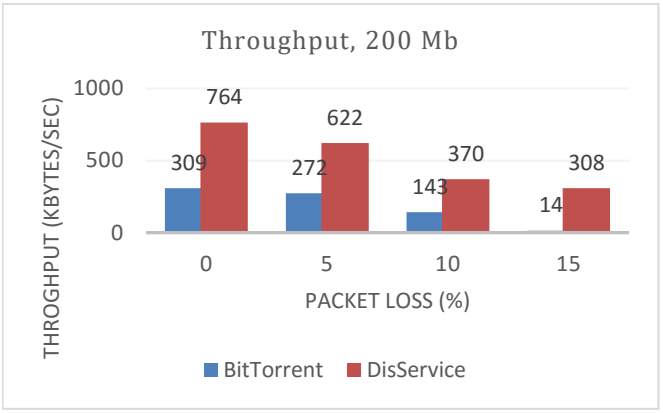

Figure 3 - Throughput measured transferring a 200 MB file

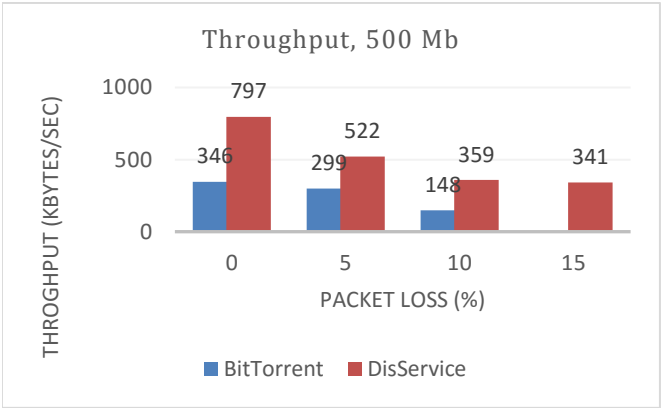

Figure 4 - Throughput measured transferring a 500 MB file

as opposed to unicast, are also highlighted by the bandwidth utilization ratios presented in Table 1. Table 1 also shows that, while both approaches are sensitive to the link degradation,

DisService's bandwidth usage increases more quickly than BitTorrent's, and therefore the ratio between corresponding values of bandwidth usage increases. The reason for this faster increase is the use of SNACK. When the reliability of the link decreases, more packets are lost, and therefore the list of missing packets contained in the SNACK increases its size. Conversely, the size of the TCP's cumulative acknowledgment message is almost insensitive to the packet loss rate.

TABLE 1 - BANDWIDTH USAGE RATIO DISSERVICE/BITTORRENT

DISSERVICE / BITTORRENT	0% PKT LOSS	5% PKT LOSS	10% PKT LOSS	15% PKT LOSS
50 MB	0.05	0.05	0.08	0.09
200 MB	0.11	0.11	0.15	0.17
500 MB	0.13	0.14	0.14	N/A

VI. RELATED WORK

Due to the emerging need for P2P file sharing solutions in infrastructure-less mobile networking scenarios, several studies evaluated the performance of running P2P applications on top of MANETs.

One possible approach consists deploying an existing P2P protocol over a routing protocol for MANETs. In [6], the authors study this solution by developing a Gnutella-based P2P protocol and testing its performance over three different routing protocols for MANETs: the Destination-Sequenced Distance-Vector Routing (DSDV), the Dynamic Source Routing (DSR) protocol, and the Ad-hoc On Demand Distance Vector (AODV). From their study, it is clear that the network and the application characteristics have a large impact on the performance of the solution.

Ding and Bhargava perform an analytical analysis of several solutions aimed at enabling P2P file sharing over MANETs [7]. The study takes into account the properties of complexity, scalability, implementation effort, maintenance effort, and energy efficiency of the solution. The authors conclude by arguing that cross-layer solutions perform better than their layered counterparts, as the former can take advantage of the commonalities of P2P systems and MANETs and (partially) avoid the overhead necessary to maintain both the routing table and the P2P overlay network. Other works derive similar conclusions, such as [8], which also extends the analysis by comparing unstructured P2P protocols (based on broadcasts and flooding, such as Gnutella) against structured P2P protocols (based on a DHT). The results justify the use of DHT-based protocols only in presence of limited mobility and a high number of nodes. In accordance with these results, DisService implements an unstructured protocol, in order to cope with high node mobility and remain efficient even when the number of neighbors is small. Additionally, since DisService does not build an overlay network between peering nodes, it achieves similar benefits to those provided by cross-layer P2P solutions.

A few research efforts have evaluated the performance of the BitTorrent protocol over MANETs. The work of Rajagopalan and Shen proposes an adaptation of BitTorrent for MANETs called BTM and compares it against a standard BitTorrent implementation called BTI [9]. BTM removes any dependencies from a central tracker and it favors connections that, in average, have shorter path lengths than BTI. However, BTM still relies on TCP to transfer file pieces, which is not disruption tolerant and suffers severe problems due to poor bandwidth utilization in MANETs [10]. CodeTorrent solves this issue by switching to UDP and allowing only one-hop transfers of file pieces [11]. This approach is effective in highly

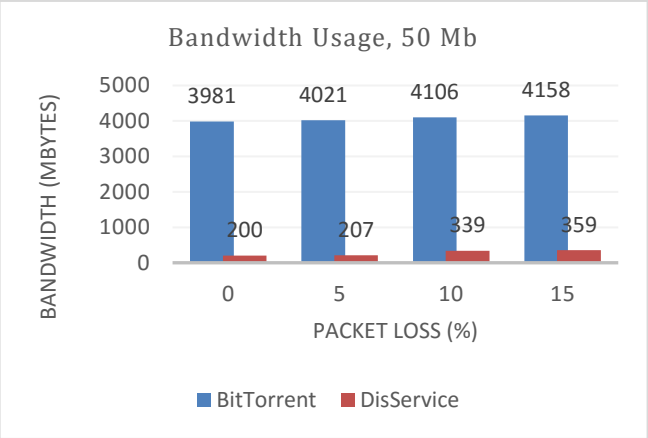

Figure 5 - Bandwidth usage measured transferring a 50 MB file

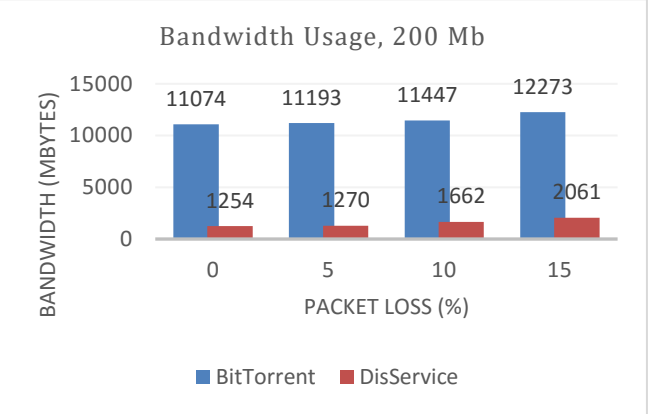

Figure 6 - Bandwidth usage measured transferring a 200 MB file

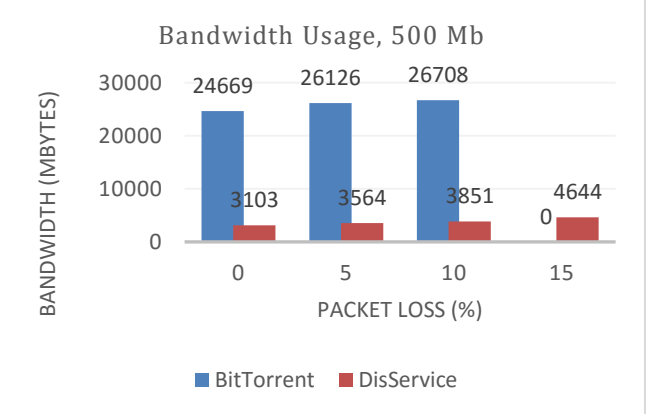

Figure 7 - Bandwidth usage measured transferring a 500 MB file

dynamic networks, such as VANETs, but it limits file sharing to the set of peers in the transmission range of each node. Sisto [12] is another adaptation of the BitTorrent protocol for

MANETs that eliminates the need for a central tracker node by creating a local ad-hoc DHT overlay network and exploits information on the network status to perform peer selection. However, Sisto requires a set of known addresses stored locally to enable nodes to join the DHT overlay network. Additionally, the authors do not specify the transport protocol that Sisto uses to transfer file segments.

VII. Conclusions and Future Work

In this paper we compared the transmission performance of DisService and Ttorrent. DisService is a dissemination service specifically designed for TENs, while Ttorrent is a CoT implementation of the popular BitTorrent protocol.

The experiments show how DisService outperforms BitTorrent in all the emulated scenarios. Moreover, the rate at which the throughput decreases clearly confirms that TCP is not a feasible choice in environments with unreliable links.

The scenario presented in this paper was purposely kept simple, even if possibly not representative of a real world scenario where the reliability of the links is likely to change over time depending on the environment and the movement of the nodes. However, its simplicity allows for easier interpretation of the resulted performance of the two protocols. Currently, we are developing more realistic scenarios that use links with variable reliability and delay. Finally, while we believe that many-cast approaches as the one implemented in DisService are in general more advantageous then approaches based on unicast, we are planning on comparing DisService to modifications of BitTorrent that do not rely on TCP for file transfer, such as CodeTorrent [11].

References

[1] P. K. Gummadi, S. Saroiu, and S. D. Gribble, “A measurement study of peer-to-peer file sharing systems”, in Proceedings of Multimedia Computing and Networking, San Jose, CA, USA, 2002, pp. 156-170.

[2] J. A. Pouwelse, P. Garbacki, D. H. J. Epema, and H. J. Sips, “The BitTorrent P2P file-sharing system: measurements and analysis”, in 4th International Workshop, IPTPS 2005, Ithaca, NY, 2005, pp. 205-216.

[3] H. Schulze, K. Mochalski, “Internet Study 2008, 2009”, Leipzig, Germany, 2011.

[4] N. Suri, G. Benincasa, S. Choy, S. Formaggi, M. Gilioli, M. Interlandi, J. Kovach, S. Rota, R. Winkler, "Disservice: A peer-to-peer disruption tolerant dissemination service", IEEE Military Communications Conference (MILCOM), 2009.

[5] N. Suri, G. Benincasa, "Opportunistic Listening System and Method, US Patent 8493902, 2013.

[6] L. Barbosa e Oliveira, I. G. Siqueira and A. A. Ferreira Loureiro, “Evaluation of ad-hoc routing protocols under a peer-to-peer application”, in Wireless Communications and Networking Conference, 2003.

[7] G. Ding and B. Bhargava, "Peer-to-peer file-sharing over mobile ad hoc networks", in Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, 2004, pp. 104-108.

[8] M. Bisignano, G. D. Modica, O. Tomarchio and L. Vita, "P2P over Manet: a comparison of cross-layer approaches", in 18th International Workshop on Database and Expert Systems Applications (DEXA 2007), Regensburg, 2007, pp. 814-818.

[9] S. Rajagopalan and C. C. Shen, "A Cross-layer Decentralized BitTorrent for Mobile Ad hoc Networks," in 3rd Annual International Conference on Mobile and Ubiquitous Systems - Workshops, San Jose, CA, 2006, pp. 1-10.

[10] X. Chen, H. Zhai, J. Wang, and Y. Fang, "TCP performance over mobile ad hoc networks”, in Canadian Journal of Electrical and Computer Engineering, Vol. 29, No. 1/2, pp. 129-134, 2004.

[11] U. Lee, J.-S. Park, J. Yeh, G. Pau, and Mario Gerla, “CodeTorrent: Content Distribution using Network Coding in VANET”, in Proceedings of the 1st international workshop on Decentralized resource sharing in mobile computing and networking, pp. 1-5, 2006.

[12] K. Manousakis, S. Eswaran, D. Shur, G. Naik, P. Kantharaju, W. Regli, and B. Adamson, “Torrent-Based Dissemination in Infrastructure-Less Wireless Networks”, Journal of Cyber Security and Mobility, Vol. 4, No. 1, Jan. 2015.

[13] Mobile Ad-Hoc Network Emulator (MANE), available online at: http://cs.itd.nrl.navy.mil/work/mane/index.php

[14] Ttorrent, a pure-Java implementation of the BitTorrent protocol: https://github.com/mpetazzoni/ttorrent

[15] Carter, C., S. Yi, P. Ratanchandani, and R. Kravets “Manycast: Exploring the Space between Anycast and Multicast in Ad hoc Networks”, in Proceedings of the 9th annual International Conference on Mobile Computing and Networking, NY, USA 2003.

[16] L. Liu, Y. Jing, Y. Zhang, B. Xia, "A Survey on P2P File Sharing Algorithms over MANETs", in Consumer Electronics Times, Vol. 2, No. 2, Apr. 2013, pp. 109-115.

[17] D. Neves da Hora, D. R. Macedo, J. M. S. Nogueira, and G. Pujolle, "Optimizing Peer-to-Peer content discovery over wireless mobile ad hoc networks", in the 9th IFIP International Conference on Mobile Wireless Communications Networks, Cork, 2007, pp. 6-10.

[18] Optimized Link State Routing Protocol (OLSR), RFC 3626, Oct. 2003, available online at: https://tools.ietf.org/html/rfc3626.

[19] Ad hoc On-Demand Distance Vector (AODV) Routing, RFC 3561, Jul. 2003, available online at: https://tools.ietf.org/html/rfc3561.

[20] G. Benincasa, A. Morelli, C. Stefanelli, N. Suri, and M. Tortonesi, "Agile Communication Middleware for Next-Generation Mobile Heterogeneous Networks”, in IEEE Software, vol. 31, no. 2, Mar.-Apr. 2014, pp. 54-61.