Accelerating File Downloads in Publish Subscribe Internetworking with Multisource and Multipath Transfers

Yannis Thomas, Christos Tsilopoulos, George Xylomenos, George C. Polyzos
Mobile Multimedia Laboratory
Department of Informatics
Athens University of Economics and Business,
Athens, Greece
Email:{thomasi, tsilochr, xgeorge, polyzos}@aueb.gr

***Abstract*—We present mmFTP, a file transfer protocol for the Publish Subscribe Internetworking (PSI) architecture, which follows the Information-Centric Network (ICN) paradigm. mmFTP is designed to utilize diverse in-network resources: (i) it is receiver-driven, thus supporting on-path caching, (ii) it downloads files from multiple sources, thus utilizing off-path caching and (iii) it offers multipath transfers, thus exploiting path diversity and assisting network load-balancing. mmFTP combines these features into a single framework without complicating network operation. This is achieved by exploiting the functional organization of the PSI architecture which - among other aspects - separates routing from packet forwarding, delegates routing control to a logically centralized module and employs an explicit routing scheme for packet forwarding. In this paper we introduce the basic operation of mmFTP and present preliminary experimental performance results from a prototype implementation deployed in the PlanetLab testbed.**

I. Introduction

One of the driving forces in *Information Centric Networking* (ICN) research is the design of architectures and protocols that efficiently utilize network resources for content delivery. In addition to *communication*, ICN brings *data storage* and *computation* to the spotlight of available network resources. Most data delivery mechanisms proposed for ICN exploit caching, either *on-path* for packets cached in routers (short-term memory) or *off-path* when entire content-objects reside in dedicated caches (long-term memory). Experience from content-distribution applications and protocols however indicates that content distribution can also benefit from multisource [1] and multipath [2] transfers, i.e. the use of multiple sources and multiple paths to each source, respectively.

In this paper, we present and evaluate the *Multisource and Multipath File Transfer Protocol* (mmFTP) [3] for the *Publish Subscribe Internetworking* (PSI) ICN architecture [4]. mmFTP is designed to utilize *all* types of network resources by combining well-known content-distribution techniques into a single protocol. mmFTP is receiver-driven and supports on-path caching, thus it utilizes the network's short-term memory; as it downloads files from multiple sources, it also utilizes the network's long-term memory. Last, but not least, mmFTP supports multipath delivery, i.e. a transfer may use multiple paths to transport content from each source, thus better utilizing the available bandwidth and improving network load balancing.

An important aspect of our work is that mmFTP supports all these features without complicating network operation. mmFTP does not require extending PSI with complex signaling or router operation. This is due to the decoupling of content resolution, path formation, and packet forwarding functionalities in PSI [5], along with the adoption of explicit routing for packet forwarding [6].

This paper focuses on the basic operation and performance of mmFTP. We describe the signaling for sending requests to multiple sources through multiple paths, emphasizing the simplicity of doing so within PSI. We implemented mmFTP in Blackadder, the PSI prototype implementation [7], and deployed it in the PlanetLab testbed. We present preliminary performance evaluation results that demonstrate the effectiveness of multisource and multipath delivery in the unpredictable PlanetLab environment, extending the first results reported in [3].

The remainder of the paper is organized as follows. In Section II we briefly describe the PSI features that allow us to realize mmFTP while keeping network complexity low. In Section III we describe the operation of mmFTP, focusing on the signaling required to locate content sources and available paths. In Section IV we present preliminary performance evaluation results based on our implementation. Finally, we provide our conclusions and discuss future work in Section V.

II. The PSI architecture

In this section we briefly describe the PSI architecture, emphasizing the functional structure of PSI, which decouples name resolution from path formation and packet forwarding. This approach allows mmFTP to easily support multisource and multipath transfers.

A. Functional organization

PSI treats content objects as publications, content sources as publishers and content consumers as subscribers. Users are provided with a publish/subscribe API for announcing and requesting information.

A fundamental design tenet in PSI is the clear separation of its core functions: (i) the *Rendezvous* function tracks available publications and resolves subscriptions to publishers, (ii) the *Topology Management* and *Path Formation* function monitors the network topology and forms forwarding paths and (iii) the *Forwarding* function undertakes packet forwarding [5]. Network nodes in a PSI network are classified into *Rendezvous Nodes* (RNs), *Topology Managers* (TMs) and *Forwarding Nodes* (FNs), as shown in Figure 1. The implementation of the PSI core functions depends on the environment, e.g. a wide-area or a mobile ad-hoc network. We describe here the design solutions used for intra-domain operation; inter-domain aspects are subjects of ongoing research, for example, see [8] for an inter-domain Rendezvous scheme.

To deliver information in PSI, a publisher first announces a content object by publishing it to the Rendezvous function.[1] Subscriptions are also handled by the Rendezvous function, which locates available publishers for the requested content. Once some content sources are found, the Rendezvous function asks the Topology Management function to compute suitable paths in order to deliver the requested information. Finally, the Topology Management function hands a *Forwarding Identifier* (FID) containing an encoded form of the selected path to the publisher and instructs it to transmit the requested data.

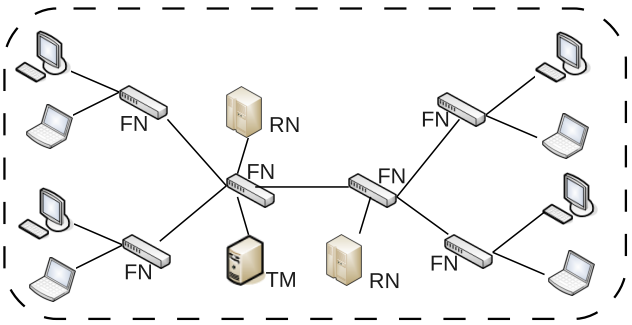


Fig. 1. A PSI network consisting of Forwarding Nodes (FN), Rendezvous Nodes (RN), a Topology Manager Node (TM) and hosts.

B. Intra-domain operation

In the intra-domain case, the Rendezvous function is jointly performed by several RNs organized in a *Distributed Hash Table* (DHT), which distribute the load of tracking publications and serving subscriptions. One or more TM nodes maintain an up-to-date view of the network topology by gathering link-state information directly from the FNs. When Rendezvous occurs, the RN that resolved a subscription asks its TM to select an appropriate forwarding path between the publisher and the subscriber and to create the corresponding FID.

For packet forwarding, PSI employs LIPSIN [6], an efficient explicit-routing scheme based on Bloom filters. In LIPSIN, each network link is labelled with a long bit string produced by a set of hashing functions. In order to create the FID for a route, we OR the labels of all path links. The resulting Bloom filter is inserted in the packet header. To test whether a packet should be forwarded over a link, we AND the Bloom filter in the path header with the link label and check whether the result is the same as the label; if so, the packet is forwarded.

After the LIPSIN FID is formed, the TM hands it to the publisher and asks it to transmit the requested publication over the specified path. In mmFTP, we take advantage of explicit-routing for multipath communication, by having the TM select multiple paths, from one or more publishers to the subscriber, and then construct the respective FIDs.

[1]In PSI terminology, *publishing* is equivalent to an *advertisement* and does *not* involve the transmission of data.

III. Multisource and Multipath file transfers in PSI

A. Design goals

Our primary goal in mmFTP is to utilize *all* available network resources in the ICN context, i.e. *communication*, *data storage* and *computation*. mmFTP achieves this goal via the following design choices:

- **Receiver-driven operation**: the receiver (subscriber) coordinates the data transmission by sending requests for individual data packets to the source node (publisher). Packet requests are forwarded directly to the publisher but may also be served by any on-path router that has a cached copy of the requested packet, thus utilizing the network's short-term memory (packet caches in routers).
- **Multisource downloads**: the requested file is retrieved from multiple locations simultaneously by sending requests to multiple publishers. This utilizes both long-term network memory (caching nodes) and available bandwidth, since distinct dissemination routes are used.
- **Multipath delivery**: if there are multiple paths between the receiver and a specific source, mmFTP further utilizes the available bandwidth resources by sending packet requests and receiving data packets via all those paths.
- **Centralized path selection**: mmFTP relies on PSI's Rendezvous function for locating multiple sources and the Topology Management function for computing available paths, so as to utilize in-network computation resources.

B. mmFTP operation

mmFTP operation is split in two phases: (i) *slow-path* rendezvous, which deals with service establishment, and (ii) *fast-path* rendezvous, which deals with the immediate host interaction for content delivery (Figure 2).

1) Slow-path Rendezvous: Initially, content sources publish and receivers subscribe to the desired file. The subscriptions are routed via the DHT to the corresponding RN (Fig. 2(a) and step 1 in Fig. 2(b)) where they are matched with the publications. When a match occurs, the RN requests the TM to compute paths between the aforementioned publishers and the subscriber (step 2 in Fig. 2b)). The TM, having a complete view of the network, computes multiple paths for each source-receiver pair and constructs the LIPSIN FIDs for both the *reverse* direction, i.e. subscriber-to-publisher(s), and the *forward* direction, i.e. publisher(s)-to-subscriber. Finally, it sends the FIDs to the subscriber (step 3 in Fig. 2(b)).

At this point, the subscriber has obtained two sets of source-routes. The former points to publishers holding the requested file (reverse FIDs) and the latter will route data from the publishers to the requesting host (forward FIDs). Note that there is no strict one-to-one mapping between source-routes and publishers; some FIDs may point to the same source if the TM decided to use the multipath capability for that particular source.

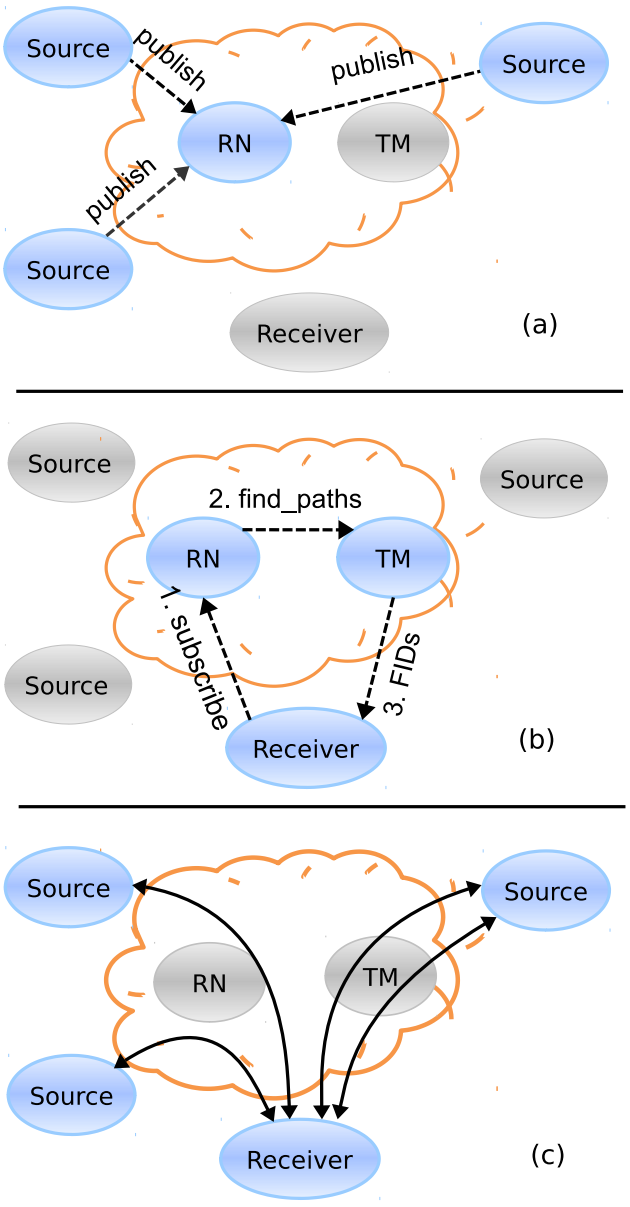


Fig. 2. mmFTP operation. (a) Content sources publish file availability to the Rendezvous function. (b) Slow-path rendezvous: receiver subscribes to file and receives list of FIDs. (c) Fast-path rendezvous: receiver sends packet subscriptions directly to sources using the obtained FIDS.

2) Fast-path Rendezvous: The receiver starts sending subscriptions for individual data packets to the publisher(s) using the reverse FIDs (Fig. 2(c)). These are fast-path subscriptions, sent directly to the sources, bypassing the DHT. Each request also carries the forward FID that encodes the path that should be taken by the response. When the subscriptions reach the publishers, the requested data packets are transmitted back to the subscriber using that forward FID.

A fast-path subscription may also be served by an on-path cache; an FN equipped with a cache can inspect fast-path subscriptions and respond if the requested packet is locally stored.[2] Note that in mmFTP sources are stateless: they just respond with the requested data packets.

C. Error control

In mmFTP, error control is applied by the receivers. For each fast-path subscription, the receiver maintains a timer. If the requested data packet does not arrive on time, suggesting that either the request or the data packet was lost, the subscriber re-issues the subscription. Since fast-path subscriptions are self-contained, this subscription may be sent either over its previous path or over a different one.

D. Flow and congestion control

Data flow is controlled by the subscriber. Upon the receipt of the LIPSIN FIDs, the subscriber creates a distinct *sub-flow* for each supplied path. Each sub-flow uses an independent sliding window that defines the maximum number of on-the-fly packets at any time over its own path. The size of the window is managed according to the standard TCP *slow start* and *congestion avoidance* algorithms [9]. Thereupon, sub-flows request packets as their congestion window slides or increases. Packet requests follow an ascending order (from the first to the last): sub-flows request the next idle packet, i.e. a packet that has not been requested by another sub-flow.

The actual decision to increase/decrease a sub-flow's congestion window affects the TCP friendliness of the protocol and is subject to the congestion control mechanism implemented. In [2], [10] the authors argue that friendliness to other flows is a crucial requirement for a general purpose transport protocol; the increasing popularity of greedy protocols, such as BitTorrent, indicates that this is not universally respected. It is important to note that due to LIPSIN's explicit routing and TM's centralized path selection, another possibility is to use rate-controlled schemes over appropriately traffic-engineered paths.

ICN brings to the table another promising feature: *in-network* congestion control. At the moment, there is an ongoing discussion in the ICN research community on whether congestion control should be applied solely at the communicating endpoints [10] or whether network routers should play a role in congestion control [11]. In our case, a hybrid co-operative model is possible, where the network (via the TM) notifies the receiver about path formation (specifically, about shared bottlenecks) and the receiver applies congestion and flow control. Specifically, when the paths are completely disjoint (hence, TCP friendliness is not an issue) the receiver can treat each sub-flow independently. If, on the other hand, the paths share a network resource, the receiver can couple all

[2]The explicit-routing scheme used allows the forward FID to be used by any intermediate node on the path to reach the subscriber.

sub-flows, so as to ensure that they are no more aggressive than a single-flow connection.

In this paper we focus on the establishment of a multi-flow connection, emphasizing the simplicity of the mmFTP design due to PSI's structure. Due to space limitations, we do not describe in detail the congestion control algorithm used. However, mmFTP, follows the principles of *mptcp* [2], using a friendliness factor *m* to control the increase of each subflow's window. The evaluation of mmFTP when using the hybrid congestion control scheme suggested above is deferred to future work.

E. Multisource and Multipath made easy

mmFTP combines well-known content distribution techniques into a single transport protocol: multisource downloading is widely adopted by P2P applications [1] and multipath transfer is a well-established research topic [2], [12]. What mmFTP adds is the exploitation of the PSI architecture and the LIPSIN explicit-routing scheme so as to not only support multisource and multipath transport, but also keep network operation simple, provide a generic interface for content delivery and utilize network storage. We now explain how these aspects of PSI simplify multisource and multipath transport.

First, the separation of the core network functions in PSI along with the choice of explicit-routing results in simple signaling and stateless FNs. Routing is orthogonal to forwarding: FNs operate in a stateless manner by using in-packet LIPSIN FIDs, without routing state, knowledge of the actual data path or the transfer state. These gains are achieved in exchange for the additional delay required at the slow-path rendezvous phase, where the initial subscription is resolved and the TM computes suitable data paths. We acknowledge that the centralized nature of path computation in the TM raises scalability concerns. Yet, we believe that TM functionality will be placed in nodes with enhanced capabilities, hence we view this as a way of utilizing in-network computation resources. In addition, this design is well-aligned with on-going developments in *Software Defined Networking* (SDN).

Second, the separation of routing and forwarding allows the transparent implementation of multisource and multipath services. The FIDs that the TM delivers to the endpoints are a set of distinct "options" for requesting data. These options may involve different publishers and/or paths, but this information is concealed from the hosts. The subscriber evaluates in real-time the performance of each option (i.e. path) and adjusts the amount of data to be delivered through it accordingly. Hence, mmFTP provides a generic interface, transparently supporting any combination of multisource and/or multipath services. The nature of the service is decided by the TM, according to a network domain's policies and goals. Essentially, the TM selects the paths, the FNs realize those paths and the subscriber controls each path's utilization.

Third, explicit-routing and centralized path selection assure diversity of transmission routes and friendliness towards single-flow connections. Multipath protocols are commonly compromised by IP's hop-by-hop routing, which can force paths from multi-homed hosts to converge over the same bottleneck links. In contrast, in PSI the FIDs indicate dissemination paths that may be selected to avoid convergence at bottleneck links. At the same time, TCP-based multisource solutions such as BitTorrent, are not friendly towards single-flow TCP when operating over the same link, while others, like MTCP, throttle bandwidth aggregation in order to avoid starving standard TCP flows [2]. In PSI, path formation may select only disjoint paths, thus allowing each sub-flow to operate over distinct links. Alternatively, similarly to mmFTP's choice, the TM informs the subscriber when many sub-flows exploit the same network resource, so as to appropriately tune its congestion window management. All in all, there are several ways to enforce TCP friendliness among end-to-end connections in PSI, undertaken either by the TM or by the end-hosts.

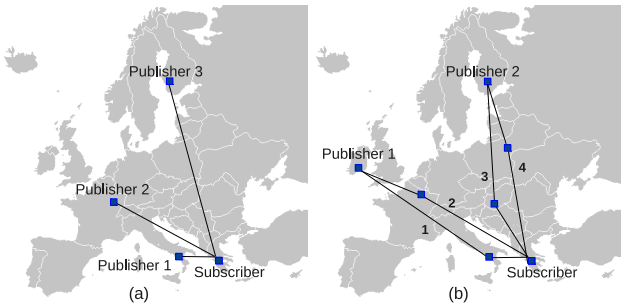


Fig. 3. PlanetLab overlay topologies.

IV. Implementation and Experimentation

A. Implementation

We implemented mmFTP over Blackadder, the PSI prototype implementation [7]. Our implementation includes the mmFTP[3] sender and receiver applications, as well as a TM that can compute multiple paths between two nodes. Our TM computes the k-shortest paths from every publisher towards the subscriber, using the algorithm by Yen [13] with hop count as the metric.

B. PlanetLab deployment

For our experiments, we deployed Blackadder with our mmFTP implementation on the PlanetLab testbed. We chose PlanetLab in order to evaluate our design in a realistic environment with actual propagation delays, forwarding overhead and competing traffic. The deployment is realized as an overlay network: a set of Blackadder nodes scattered across Europe (Figure 3), communicating via UDP tunnels. We examined two network topologies and several transfer schemes, so as to investigate the gains from multipath and multisource content delivery in terms of performance, resilience, and load balancing. Since the overlay topology can conceal several shared bottlenecks, we did not exploit the TM's knowledge of dissemination paths. mmFTP achieves TCP friendliness by occupying bandwidth with equal rate to a single-flow

[3]An open source implementation of mmFTP is available at http://mm.aueb.gr/research/mmFTP/

connection that uses the slow start and congestion avoidance algorithms.

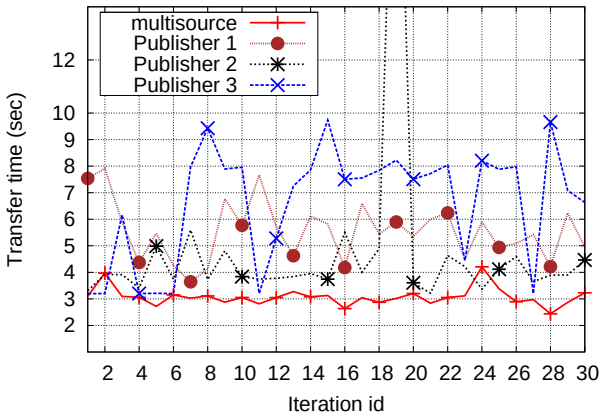


Fig. 4. Download times with single/multi-source transfers.

C. Preliminary evaluation results

1) Performance gains with multisource: Our first scenario examines the bandwidth gains that can be achieved when a subscriber downloads a 12 MB file from 3 publishers, using the topology in Fig. 3(a). The experiment consists of four phases: the subscriber (located in Greece) first downloads the file in single-source mode from each publisher in turn, and then it downloads the file from all three publishers in multi-source mode. As congestion in PlanetLab is unpredictable, we performed 30 iterations of the experiment, resulting in 120 downloads. Figure 4 shows the download time in each iteration. The best performance corresponds to the multisource case, with an average download time of 3.07 s (equivalent to 3.9 MB/s). The best single-source performance is achieved with Publisher 2 (in France), with an average download time of 4.8 s (equivalent to 2.5 MB/s).

As evidenced by the spikes in Fig. 4, mmFTP is much more stable in multisource mode: the variance of the download times for our 30 iterations was only 0.1 in multisource mode, while in single-source mode the variances were 1.13, 15.75 and 4.16 for Publishers 1, 2 and 3, respectively. This is due to the adaptation of the receiver to the prevailing network conditions: mmFTP dynamically avoids paths that exhibit congestion, a situation that we often met in PlanetLab.

2) Resiliency to node/path failures: Our second scenario investigates mmFTP's robustness to path failures. We downloaded a 50 MB file in multisource mode using the topology of Fig. 3(a) and emulated path failure by shutting down Publisher 2 during the transfer. Specifically, Publisher 2 was programmed to stop responding to all packet requests at a certain time, remain idle for 7 seconds and then return to normal operation. To assess the impact of path failure to mmFTP, each iteration of the experiment consisted of one download with path failure and one without, and we performed 20 iterations.

Figure 5 shows the evolution of the average congestion window size to each publisher over time in the *normal* and *failure* mode. In normal mode, the file is downloaded from all three sources in 12.2 s (equivalent to 4.09 MB/s). In failure mode, Publisher 2 fails at approximately 20% of the transfer duration and resumes at approximately 60%. During that time interval, the mmFTP receiver automatically switches to Publishers 1 and 3, increasing their download rates according to their path capacities: the rate of Publisher 1 is increased by 76.5%, while the increase observed at Publisher 3 is approximately 30%. As a result, the average download time in failure mode increases by only 2.1 s to 14.3 s (equivalent to a 0.6 MB/s drop to 3.49 MB/s), despite a failure in the highest capacity path (to Publisher 2) during half of the transfer.

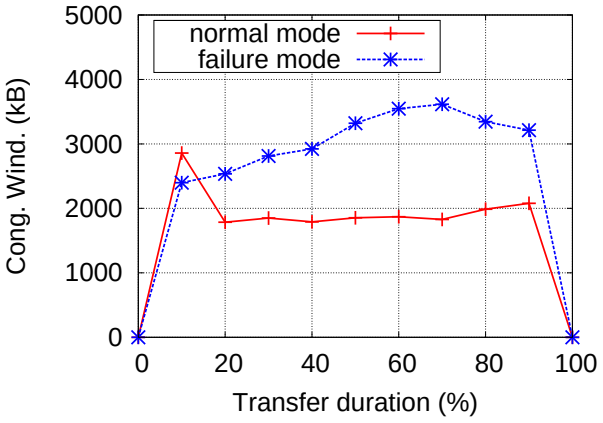
3) Performance gains with multipath: Our third scenario investigates the additional performance gains due to multipath transmission. For this scenario we used the topology shown in Fig. 3(b), where the receiver can use two disjoint (overlay) paths towards each of the two available sources. In each experiment the receiver first downloaded a 50 MB file in multisource mode using a single path per source (paths 1 and 3), and then it downloaded the same file in multisource and multipath mode, exploiting all four paths, repeating this pattern 10 times (20 downloads). We repeated this entire experiment 4 times, in order to show how unpredictable PlanetLab is, even when we aggregate multiple measurements.

Figure 6 shows the average download times for each repetition. The average download time with multipath was reduced by 31.5%, 10.4%, 19.9% and 11.6%, respectively, for an overall gain of 17%. This is due to mmFTP's ability to effectively avoid bottleneck links, utilizing the least congested paths. From our experience with PlanetLab, which suffers from large bursts of congestion, the exploitation of *backup* paths allows traffic to be switched as needed, hence multipath transfers lead to faster downloads. In addition, multipath transfers exhibit less variation than single-source ones, a pattern also observed in the single-source vs. multisource comparison, further increasing the stability of mmFTP.

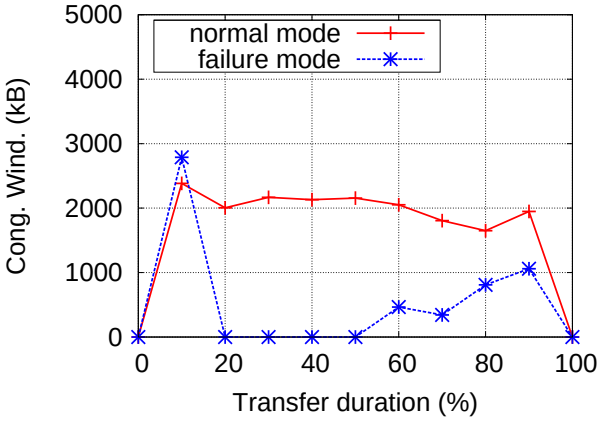
V. Conclusions and Future Work

We presented mmFTP, a receiver-driven file transfer protocol for the PSI architecture supporting multisource and multipath transfers. mmFTP combines well-known content distribution techniques in a single protocol, without requiring complicated network signaling or adding state to routers, due to the design choices of the PSI architecture. We implemented a prototype of mmFTP and evaluated its performance on PlanetLab. Our results verify the effectiveness of both multisource *and* multipath delivery, in terms of both throughput and stability.

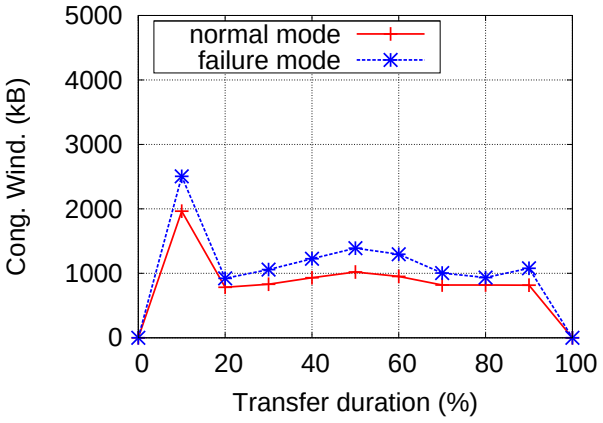
While mmFTP constitutes a promising framework for ICN file transfers, many aspects of its operation require further research. One issue is how sources and data paths can be selected by taking into account network state and the location of content sources. Another issue is the design of congestion control schemes for multisource and multipath transfers that take advantage of PSI's capabilities and their evaluation with respect to friendliness and effectiveness. In the case of in-network congestion control schemes, issues related to fea-



(a) Publisher 1



(b) Publisher 2



(c) Publisher 3

Fig. 5. Evolution of the average congestion window size to each source in normal and failure mode.

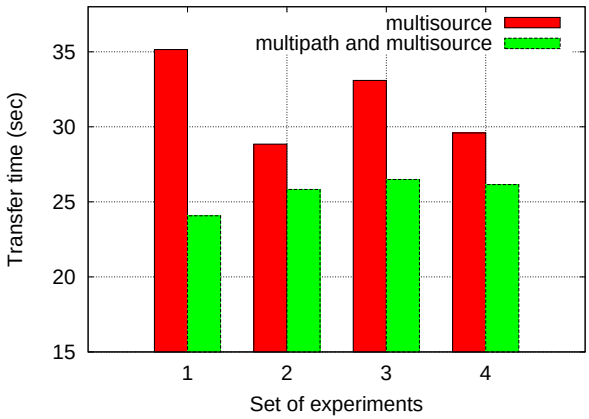


Fig. 6. Average download times with/without multipath.

sibility and scalability need to be examined along with the performance aspects.

ACKNOWLEDGEMENT

This research has been co-financed by the European Union (European Social Fund-ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF)-Research Funding Program: Aristeia II/I-CAN.

REFERENCES

[1] Androutsellis-Theotokis, S.: Spinellis, D.: A survey of peer-to-peer content distribution technologies. ACM Comput. Surv. Vol. 36, No. 4, pp. 335 - 371, 2004.
[2] Wischik, D.: Raiciu, C.: Greenhalgh, A.: Handley, M.: Design, implementation and evaluation of congestion control for multipath TCP. Proc. of the USENIX NSDI Conference, 2011.
[3] Thomas, Y.: Tsilopoulos, C.: Xylomenos, G.: Polyzos, G.P: Multisource and multipath file transfers through publish-subscribe internetworking. Proc. of the 3rd ACM ICN Workshop, 2013.
[4] Xylomenos, G.: Vasilakos, X.: Tsilopoulos, C.: Siris, V.: Polyzos, G.C.: Caching and mobility support in a publish-subscribe Internet architecture. IEEE Comm. Magazine. Vol. 50, No. 7, pp. 52 - 58, 2012.
[5] Trossen, D.: Sarela, M.: Sollins, K.: Arguments for an information-centric internetworking architecture. ACM SIGCOMM Comput. Commun. Rev. Vol. 40, No. 2, pp. 26 - 33, 2010.
[6] Jokela, P.: Zahemszky, A.: Esteve-Rothenberg, C.: Arianfar, S.: Nikander, P.: LIPSIN: line speed publish/subscribe inter-networking. Proc. of the ACM SIGCOMM Conference, 2009.
[7] Trossen, D.: Parisis, G.: Designing and realizing an information-centric Internet. IEEE Comm. Magazine, Vol. 50, No. 7, pp. 60 - 67, 2012.
[8] Katsaros, K.: Fotiou, N.: Vasilakos, X.: Ververidis, C.: Tsilopoulos, C: Xylomenos, G.: Polyzos, G.C.: On inter-domain name resolution for information-centric networks. Proc. of IFIP Networking, 2012.
[9] Jacobson, V.: Congestion avoidance and control, In ACM SIGCOMM Comput. Commun. Rev. Vol. 18, No. 4, pp. 314-329, 1988.
[10] Carofiglio, G.: Gallo, M.: Muscariello, L.: ICP: Design and evaluation of an interest control protocol for content-centric networking. Proc. of the IEEE NOMEN Workshop, 2012.
[11] Rozhnova, N.: Fdida, S.: An effective hop-by-hop interest shaping mechanism for CCN communications. Proc. of the IEEE NOMEN Workshop, 2012.
[12] Hopps, C.: Analysis of an equal-cost multi-path algorithm, RFC 2992, 2000.
[13] Yen, J: Finding the k shortest loopless paths in a network, Science Management. Vol. 17, No. 11, pp. 712 - 716, 1971.