Towards Peer-to-Peer Content Retrieval Markets: Enhancing IPFS with ICN

Onur Ascigil
University College London
o.ascigil@ucl.ac.uk

Sergi Reñé
University College London
s.rene@ucl.ac.uk

Michał Król
University College
London/UCLouvain
michal.krol@uclouvain.be

George Pavlou
University College London
g.pavlou@ucl.ac.uk

Lixia Zhang
UCLA, USA
lixia@cs.ucla.edu

Toru Hasegawa
Osaka University
t-hasegawa@ist.osaka-u.ac.jp

Yuki Koizumi
Osaka University
ykoizumi@ist.osaka-u.ac.jp

Kentaro Kita
Osaka University
k-kita@ist.osaka-u.ac.jp

ABSTRACT

In the current Internet, content delivery, *e.g.,* video-on-demand (VoD), at scale is associated with a large distributed infrastructure which requires considerable investment. Content Providers (CPs) typically resort to third-party Content Distribution Networks (CDNs) or build their own expensive content delivery infrastructure in order to cope with the peak demand and maintain sufficient quality-of-service (QoS), while Internet Service Providers (ISPs) need to overprovision their networks. In this paper we take a first step towards designing a system that uses storage space of users as CDN caches and deliver content with sufficient (*i.e.,* CDN-like) quality while rewarding users for their resource usage as in a *content retrieval marketplace*. As a possible candidate for such a system, we consider recent P2P storage and delivery systems that have adopted new mechanisms such as rewarding of useful work (*e.g.,* storage) while ensuring fairness and accountability through cryptographic proofs. In this paper, we experiment with the popular Interplanetary File System (IPFS) and investigate its performance in delivering VoD content locally within an ISP. Our findings suggest that operating IPFS (operating on top of IP) has its performance limitations and complementing it with an ICN network layer can significantly improve the delivery quality. We then propose and compare several forwarding strategies for ICN which can efficiently route requests and balance the load between peers with limited uplink resources.

CCS CONCEPTS

• **Computer systems organization** → **Peer-to-peer architectures**; • **Networks** → *Routing protocols*;

1 INTRODUCTION

Global Internet traffic is rapidly increasing and is expected to increase threefold over the next five years [3]. The driving force behind this growth is Video-on-Demand (VoD), which is expected to double by 2021 [3] and constitutes nearly 70% of the Internet peak traffic [4]. Because VoD content requests are by nature asynchronous, they involve unicast delivery by the *content providers (CPs)* to each individual end user.

Although CPs can manage the unicast delivery of VoD traffic through their in-house server and caching infrastructure during off-peak hours, they either resort to third-party *Content Distribution Networks (CDNs)* or build their own content delivery infrastructure

in order to cope with the demand and continue providing sufficient quality-of-service (QoS) to user applications during peak times [9]. VoD delivery at scale requires a substantial investment for a large, distributed CDN infrastructure in order to ensure timely delivery.

CDNs typically deploy massively distributed cache servers, where content is available at many locations, and they dynamically resolve (i.e., map) content requests to appropriate cache servers. Although ISPs benefit from decreasing *transit costs* resulting from reduced peak time upstream traffic volumes, they struggle to cope with rapid traffic shifts caused by the dynamic server selection policies of the CDNs, making it difficult to perform traffic engineering [36]. Although major CDNs often use measurements of network conditions within an ISP, the resolution can introduce areas of congestion within the ISP network. Furthermore, CPs with popular content are reluctant to deploy third-party CDNs, because the usage-based payment model (i.e., per-byte delivery charge) of the CDNs can be extremely costly for such CPs.

At the same time, peer-to-peer (P2P) storage and delivery is making a come back with the added features of content integrity, incentives (i.e., through rewarding), fairness in participation, trust establishment, *etc.* which the earlier P2P systems lacked. Nowadays, a plethora of decentralised P2P storage networks have emerged such as InterPlanetary File System (IPFS) [17], Storj [28] or Maid-Safe [29]. This is a result of advances in cryptography, ensuring fair exchange [21], calculating incentives reliably [18, 27], and enabling distributed payments through a blockchain [34]. As an alternative to the increasingly centralised Internet services that use back-end clouds (*e.g.,* used for content storage) [14] [41] [23] [20], these new solutions provide a distributed sharing economy marketplace.

In this paper, we investigate whether P2P content delivery within individual edge ISPs—*i.e.,* that is, content downloaded or pre-fetched by users are stored locally at user premise devices (ideally always-on equipment such as access points) and on-demand delivered to the users of the same ISP as shown in Fig. 1, where user C retrieves VoD content chunks from nearby users A and B—can be a viable replacement of CDN caches. We consider IPFS as the representative of such P2P storage and delivery systems due to its popularity. Our ultimate goal is to design a *content retrieval market* that incentivises both storage and high-quality (*i.e.,* timely) delivery and provides a comparable performance with a CDN service. In addition to the security and incentive mechanisms, IPFS uses an *information-centric* approach to storage and retrieval through self-certifying names for content. Also, IPFS together with several extensions [18] can
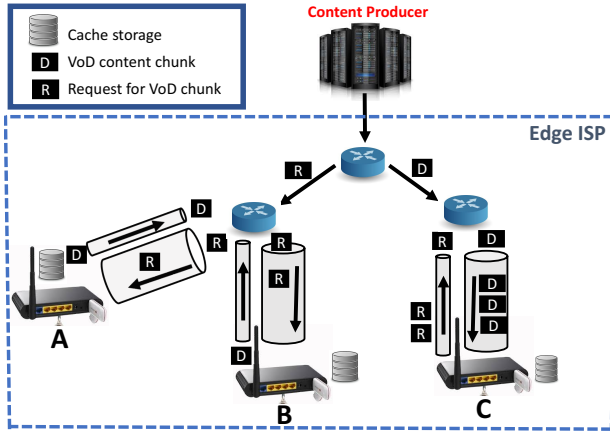
Figure 1: A P2P storage and content retrieval system.

ensure that rational users act truthful and not misbehave when providing content storage.

IPFS works as an overlay and can work on top of different underlying networking protocols, such as TCP, UDP or webRTC. However, being an application-layer protocol operating on top of an endpoint-centric IP, IPFS is limited to an application-level resolution mechanism and unicast communication. In contrast, Information-Centric Networking (ICN) solutions like NDN [45] and CCNx [1] can reduce network load by natively delivering content from cached copies of the downloaded content at the end users.

Considering VoD as the use-case, our initial finding is that IPFS creates extra overhead for the ISPs in terms of duplicate packet delivery and fails to provide high-quality delivery mainly due to its overlay nature. Furthermore, its DHT-based resolution adds extra latency which can be a significant overhead especially with small to medium size content transfers. Consequently, we propose to exploit ICN as the delivery subsystem where the ISPs have full control over the traffic. The main challenge is to design appropriate routing and forwarding mechanisms that can scale to a large number of content and provide load-aware routing strategies which balance the load among a pool of users with limited uplink bandwidth resources.

The remainder of the paper is organised as follows. We first outline the desirable features of a content retrieval system in Section 2. Then, we provide a comparison of IPFS and Named-Data Networking (NDN) architecture [45] as the representative ICN architecture in Section 3, including an initial evaluation of NDN and IPFS serving as a P2P storage and retrieval system using Docker containers (Section 4). In Section 5, we discuss how to enhance our P2P content retrieval market solution by combining IPFS and NDN. Finally, in Section 6, we provide simulations to compare several in-network name resolution mechanisms using a VoD scenario and evaluate the quality experienced by users in terms of playback buffering occurrences and overhead of the traffic on the local ISP's network.

2 A P2P CONTENT RETRIEVAL SYSTEM

In this section we describe the main requirement for a P2P Content Retrieval System implementation:

- **Data Centric:** In a content retrieval system, users do not care from where they receive the content, but rather focus on the content itself. The underlying system should reflect this approach in the design, providing content retrieval without the limitations of host-based networking, such as the inefficiencies of circuit-based retrieval, and allowing asynchronous data retrieval from any peer.
- **Scalability:** The system can consist of a large number of user and data items. Therefore, the proposed solution must scale well with increasing network size, providing data on time no matter how many nodes participate in the network or how many data items can be requested.
- **Scattered content:** Users fetch different files from the network and can decide to host the files themselves. In contrast to NDN producers being responsible for specific prefixes, P2P network users posses data under multiple prefixes fetched from different producers. The network must be able to perform efficient lookup operations without taking advantage of prefix aggregation.
- **Path Efficiency:** Linking with the data-centric requirement, the content retrieval network should provide optimal paths and deliver content from the closest available location to reduce the overhead.
- **Load Balancing:** In contrast to the cloud or massively distributed CDNs, users in a P2P network have limited uplink resources and can be easily overwhelmed with requests, as shown in Figure 1, with thin pipes for uplink and fat pipes for downlink. The network should be able to forward requests to different hosts and spread the load between them, avoiding congestion at end-users.
- **Resilience to Network Dynamics:** The system can experience high number of users continuously joining and leaving the network. Furthermore, hosts can often delete or download new files and the network must be able to efficiently handle those changes.
- **Decentralisation:** System governance should be evenly split between all the participants without privileged entities. Furthermore, the architecture can not introduce a single point of failure and should provide redundancy for all the required system components.
- **Security:** When retrieving from anonymous peers, users must be able to reliably verify the integrity of the received content. Malicious users may also launch a wide variety of attacks, including Denial-of-Service (DoS), to decrease network performance. The system should be able to mitigate those attempts and reduce their effectiveness.
- **Incentives:** To create a sustainable environment, nodes committing their resources to share content with others should be rewarded by the network (*e.g.,* by higher download priority). At the same time, selfish or malicious users should be penalised or suffer from reduced rewards.

3 ARCHITECTURAL OVERVIEW OF IPFS

There are significant differences between NDN and IPFS from an architectural point of view. While IPFS is an overlay solution that resides in the application layer and uses IP as its underlying network

architecture, NDN is a network-layer solution aimed at completely substituting IP even though it can work over IP for incremental deployability.

We believe that NDN and IPFS can complement each other in some aspects. For instance, using IP as the underlay network technology results with performance limitations that NDN can solve. Currently, IPFS needs to resolve data names to endpoint-centric, location-dependent identifiers, making the protocol translation inefficient and adding a significant delay(Section 4). In contrast, NDN does not need to resolve names due to its inherent name-based routing. On the other hand, IPFS can scale better using its DHT resolution since routing information is spread across multiple nodes.

In this section, we provide a design comparison of NDN and IPFS, analyse their ability to work together and their suitability to provide a robust peer-to-peer content delivery system.

3.1 Naming

Both architectures embrace an information-centric approach, naming data and implementing pull-based communication model. However, both naming schemes differ significantly. IPFS uses flat names consisting of self-certifying content hashes as identifiers for content objects. Such an approach makes content objects immutable, *i.e.,* changes to the contents of a data object results in a new object with a different Content ID (CID). At the same time, one can easily detect content duplicates to optimise the usage of the storage space.

In contrast, NDN uses hierarchical identifiers, but it allows the specific naming convention to be left to the applications as long as it is coherent with the security requirements, *i.e.,* content objects are signed by their producers (Section 3.3). The flexibility of NDN names means that self-certifying IPFS names may be used with specific prefix (*i.e.,* /ipfs/<content_hash>).

Name lookup and versioning: Both IPFS and NDN requires consumers to obtain the content names from a trusted source. IPFS provides a resolution mechanism where human readable names can be mapped to CIDs through DNS. Also, a separate InterPlanetary Name Space (IPNS) system enables each user to store mutable, signed records (as opposed to immutable content objects) under /ipns/<NodeId> where NodeId is the hash of the user's public key. Such mutable records are addressed by the hash of public keys, and among its other uses, they are especially useful for storing up-to-date CIDs of the latest versions of dynamic contents. The authenticity of the mutable records can be verified through the user's public key. However, IPFS can still struggle with dynamic content, as both DHT and DNS can be slow to update. On the other hand, NDN can include a version number as a component of the names and provides a DNS-like name resolution service to obtain names [11].

Collections: Apart from fetching single files, a P2P storage should enable downloading collections of files and define relations between them. IPFS implements collection files following UNIX filesystem structure and allows requesting content using relative paths within a tree-like structure (i.e., /ipfs/<collection_hash>/foo/ bar/file.mp3). Once downloaded, the collection file contains a list of all the files in the collection together with their hashes. Users can then use this information to recursively request missing content. Such an approach, allows files to be stored once, but referenced by

multiple collections under different relative names. However, once created, a collection cannot be modified, so all the collection files must be present when description is built.

In NDN, collections can be realised using naming conventions (i.e., assigning names with increasing sequence numbers to video frames) or described in manifest files[33, 42] containing lists of files in the collection as well as their checksums. However, while IPFS collections are just regular files, they can be directly requested from the network without additional support from NDN and interpreted in the application layer.

3.2 Routing and Forwarding

A P2P content retrieval system requires a way to route requests for contents to the endpoints who possess the contents in their storage. IPFS users collectively maintain a Kademlia Distributed Hash Table (DHT) [32], which maps each content name to a set of end-points. A user requesting a file must first resolve its name (*i.e.,* CID or mutable link) by querying the DHT. For fetching content, IPFS implements the BitSwap protocol which is inspired from BitTorrent. It is a message-based protocol with each message containing a list of requested chunks (*i.e.,* want list) or blocks of data. The DHT lookup reduces the number of information stored at each node, but suffers from potentially slow lookup speeds which increases with the number of nodes $n -O(log(n))-$.

In NDN, request packets (*i.e.,* Interest) carry the name of the desired content chunk which are then routed to any node who has the requested data. Interests leave "breadcrumb" state as they are forwarded, and such state is stored in the Pending Interest Table (PIT) of the forwarders. The Data packets containing the requested content chunk simply follow the breadcrumbs back to the origin of the request, following the corresponding Interest packet's path in reverse. Through the breadcrumb state in their PITs, the NDN forwarders can natively support multicast, and also store Data packets in their (in-network) caches to directly serve content to users.

NDN forwarders store forwarding information, provided by the routing protocols, in their FIB tables, and the forwarding involves longest prefix match of the content name in the Interest packets against the name prefixes in the FIB. While such an approach removes the lookup delay, it suffers from routing scalability problems because routers (in the default-free zone) must store information about all the available contents with size d. However, the hierarchical naming allows some aggregation of names (*e.g.,* /google to represent content produced by Google) to reduce routing table sizes. NDN also allows a secure mapping of content namespace to a separate namespace containing forwarding hints (*e.g.,* /att/ north-east-PoP) [13].

The routing scalability problem of NDN is even more pronounced in the P2P data distribution scenario with users providing stored content because the routing system needs to keep track of the stored content at the users. While the namespace mapping solution is applicable to this scenario, the initial resolution through NDNS also introduces additional lookup delays. When delivering Interests, *stateful forwarding strategies* are able to monitor dataplane performance (*e.g.,* current network load) and route the Interests accordingly [43]. This is an important feature in a P2P scenario

where the producers typically have limited resources and upload bandwidth. When operating with NDN, IPFS would not require to maintain a separate mapping between content and endpoints. However, DHT may still prove useful in providing forwarding hints for the network layer (Section 5).

3.3 Security

After fetching a file, users need to verify the integrity of the fetched files. Self-certifying names of IPFS allows straightforward content verification. The security model assumes that users are able to fetch the content name in advance from a trusted source. Such an approach, while simple, may be problematic when dealing with dynamically generated content and live streaming. Furthermore, file collections are secured with a Merkle DAG following the Git model. Users can thus download a single file from a collection using a single hash value and still be able to reliably verify the content. Both the file and sibling hash can be fetched from untrusted sources as long as the root of the tree is trusted.

In NDN, each data chunk includes a digital signature from its producer. When properly configured, users have trust anchors and are able to automatically download certificates of signing producer and decide whether to trust the content. Such an approach solves the problem of dynamically generated data. When maintaining IPFS names in the network layer, the application layer will still be able to keep its current security model based on self-certifying identifiers with NDN providing an additional layer of security. Furthermore, NDN provides mechanisms to manage keys and automatically bootstrap trust relations between nodes [44].

In a P2P scenario, NDN routing protocols need to advertise content stored at the users. However, untrusted users advertising content through NDN's control plane is a cause for concern because malicious users can advertise arbitrary content names that they do not possess. Consequently, NDN routers do not accept advertisements from untrusted nodes. This is an issue for the NDN routing system which requires separate mechanisms to ensure that users advertise only the content that they possess.

3.4 Incentives

For the P2P system to function properly, peers must commit their resources and provide services for others (i.e., share files). On of the main issue is thus providing incentives to motivate users to participate. When considering file exchange, numerous solutions have been proposed including centralized and decentralised reputation systems [30].

Filecoin [27], is a system built on top of IPFS that uses blockchain to reward users for storing files. It uses Proof of Replication [18] to cryptographically prove that given file was stored by user for a specified amount of time. However, while file storage can be proved, *it is currently impossible to prove a file transfer between two untrusted parties*. It means, that a malicious user can store files, claim rewards, but can either refuse to share them with other or dedicate insufficient resources when doing so. Such behaviour is also difficult to detect by IP routers without costly deep packet inspection [26].

On the other hand, stateful NDN routers can radically improve network ability to monitor and keep users accountable for their bandwidth participation. This can possibly happen through a Satisfied Interest Table that we propose in Section 5, which contains arrival times of both Interest and the corresponding Data packets in order to determine the faulty party when content is delivered with unacceptable delay. Such information can be further subitted to users-reward systems [25]. However, this is a complex problem and we leave a detailed investigation of an accountable delivery mechanism for future work.

3.5 Quality of Service

The concept of Quality of Service is still an open discussion in content based networking. Particularly, in NDN, the hop-by-hop forwarding can allow a DiffServ QoS model[24], either prioritising specific data in the cache (that can be differentiated by name[19, 37]) or by adapting the forwarding strategy to the desired QoS scheme. However there are differences between NDN and IP, such as i) no end-to-end flow, ii) the use of caches, and iii) the multi-destination, multi-path forwarding, which makes it difficult to apply the same end-to-end QoS concepts, such as bandwidth allocation or end-to-end latency.

Similarly in IPFS, it is difficult to ensure QoS because the data comes from unknown sources from arbitrary locations in the network. Moreover, there is no control over the delivery at the network level. To the best of our knowledge, the only way IPFS can provide some sort of QoS is by using Coral [22], which is a variation of the Kademlia DHT, called Distributed Sloppy Hash Table (DSHT) that uses clustering for latency-optimized hierarchical indexing. Coral successfully clusters nodes by network (latency) diameter, ensuring that nearby replicas of data can be located and retrieved without querying more distant nodes.

4 PRELIMINARY PERFORMANCE COMPARISON

In order to better understand the ability of NDN and IPFS to serve as a P2P storage and delivery network for VoD, we performed a set of initial experiments using ns-3 in emulation (real-time) mode [7]. For more realistic results, we attach docker containers (using [8]) running the production code for both projects, using IPFS version 0.4.20 and NDN Forwarding Daemon version 0.6.6. This initial evaluation consists of a simple star topology with ten peripheral nodes and a hub node, where one of the peripheral nodes act as the main server and the rest are clients as shown in Fig. 2. All the links have a bandwidth of 10 Mbps and have a propagation delay of 2 ms.

For the experiments, we emulate a VoD HTTP Live Streaming (HLS) application. Clients first download a M3U8 playlist file containing the video segments information, and then they start progressively fetching each video segment in the list until the playback buffer is full (we use 30 sec playback buffer). Once the video is consumed and there is less than 30 seconds of video in the buffer, more segments are requested. We use a 480p, 10-minute long test video [2] with a size of 27.98 MB. Users do not start all at the same time; there is a randomly generated gap of 5 to 10 seconds between subsequent users.

To evaluate NDN, we used the Network Forwarding Daemon (NFD) [5, 6] with ndn-tools *ndnputchunks* and *ndncatchunks* for the

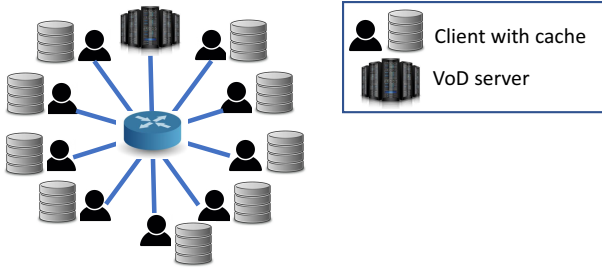|	IPFS	NDN
Naming	Content hash	Hierarchical name
Human-readable names	No	Yes
Name Lookup and Versioning	IPNS + DHT	Naming Conventions + NDNS
Collections	Collection file	Naming convention or Manifest
Routing	DHT (Kademlia)	FIB + NDNS
Routing Table Size	$O(d/n)$)	$O(d)$
Lookup speed	$O(log(n))$	$O(1)$
PDU	Bitswap Messages	Interest + Data
Security	Merkle DAG	Signatures

Figure 2: Preliminary evaluation setting.

content publication and retrieval. Both tools send Interests using the TCP-Cubic algorithm to control the window size. Once a client receives an HLS video segment from the server, it publishes the received content in the hub node so that an additional route for that segment, which points to the user that downloaded the segment, is added for future requests. We evaluated NDN using three different forwarding strategies including NCC (CCNx forwarding strategy), best-route and ASF (Adaptive Smoothed RTT-based Forwarding Strategy) [12]. To better observe the distribution of requests between nodes, we disable NDN caching on the central node.

For IPFS, we use the default Go implementation[1] and set up one node that previously adds the video segments to the network (acting as a server) while the rest of the nodes act as consumers. We keep the default configuration file, but we replace the public DHT bootstrap nodes with the node serving the video.

Figure 3(a) presents the average latency of what we call resolution time, classified by segment sizes (i.e., the elapsed time between the client requesting the video segment and actually starting to download the file). In IPFS, we measure the time necessary to resolve the hash name of the file until it is added to the want list. In NDN, we measure the time between sending of the initial interest and the receipt of the first chunk of the segment, since there is no name resolution. In contrast, with IPFS we clearly observe one of its limitations in terms of performance. IPFS needs to convert the collection hash to the hash name of each IPFS chunk through a DHT lookup. In case the video segment fits into one chunk (IPFS chunk size is 256 KB), no further resolution is required (and this initial resolution can be done only once for the whole video and cached locally). However, when the video segment is larger than one chunk, an additional recursive lookup is necessary to obtain

all the identifiers for the chunks. In the figure, we observe significantly smaller lookup times for chunks smaller than 200 KB, where the resolution time is equivalent to NDN. On the other hand, for chunks bigger than 200 KB, the resolution latency is on the order of 300-400 ms. The resolution latency is not a significant issue for VoD which performs resolution once in the beginning, but it can have a negative impact for live video content, which requires repeated resolution for upcoming segments.

In Figure 3(b), we observe the average throughput classified by segment sizes. In this figure, we observe the main issue of IPFS which is overhead on the network. The average throughput obtained using IPFS is close to a one-tenth of the NDN's. This is mainly because of the congestion created by IPFS: with IPFS, the receivers have no control of what they receive from peers—users simply register their interest in files, and the peers having the files send it out without knowing if the user already received the file from other peers, following the BitSwap protocol. Duplicate packets are later discarded at the user. We observe that this leads to unnecessary congestion which is against ISP's interest in accommodating p2p delivery for reducing the traffic in their networks.

In our evaluation, we counted a massive, 15612 duplicated chunk arrivals at the user, when approximately 1500 unique chunks are requested. This means on average nine unnecessary duplicates per chunk. This would impose a high overhead on the network. Although IPFS can parallelise content downloading once the lookup is done from multiple peers (only files bigger than one chunk can benefit from it), we argue that the performance would be significantly better using a more conservative delivery approach.

In Figure 3(c), we observe the load distribution among the nodes in the topology. IPFS is the only solution that uses all the nodes to distribute the load, even though the selection of the producers is completely random. Because most of the received packets are later discarded (i.e., duplicates), most of the content used in the playback are still provided by the first node. NDN achieves a different load distribution depending on the forwarding strategy used. The best-route approach is the one that offloads the traffic most out of the video server. However, this strategy substitutes the original route to the server once with the route to the first user that downloads the content and keeps the latter route throughout the experiment because the cost of the path between all the users are equal. The ASF strategy balances traffic based on RTT. Because there is no server congestion in these experiments and there are not many RTT changes, we observe that the strategy uses only two users.

---

[1]https://github.com/ipfs/go-ipfs

(a) Resolution time
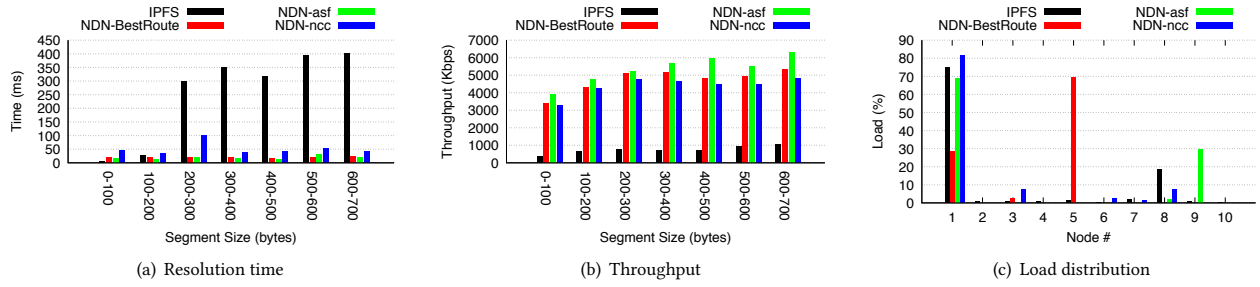
(b) Throughput

(c) Load distribution

Figure 3: Preliminary results

The NCC strategy keeps most of the traffic in the original video server, but utilises more users than ASF.

To sum up, NDN can be faster and more efficient than IFPS for VoD content retrieval, because dealing with content distribution at network instead of application level allows better control of load distribution and congestion. Furthermore, NDN can use PIT aggregation and caching to reduce the number of bytes transferred in the network. However, the amount of FIB state at the forwarders and the need for the routing protocol to advertise the stored segments at the untrusted users are important concerns. More importantly, our preliminary results demonstrate the need for a better strategy to distribute the load between the serving nodes than the inspected forwarding strategies, ideally retrieving content from the closer node but without creating congestion. Thus, in the following we explore different solutions for the aforementioned issues.

5 A P2P CONTENT STORAGE AND RETRIEVAL SYSTEM USING ICN

NDN [45] can provide a more sophisticated per-hop forwarding behaviour than IP through its use of soft-state per named packet. More specifically, NDN forwarders enforce a routing symmetry between Interest and the corresponding Data packets, which enables per-Interest dataplane performance measurements which can be used to adapt future forwarding decisions. One example is the use of round-trip time (RTT) measurements on the forwarded Interest packets by the ASF forwarding strategy.

The default forwarding behaviour in IP is to simply route Interest packets along the shortest paths with minimal load-balancing mechanisms (e.g., Equal-cost Multi-Path routing) to the closest producer with the assumption that producers have sufficient bandwidth resources to deal with whatever network throws at them in terms of requests to process and respond to. This assumption, however, does not hold in the case of P2P scenarios as users typically have limited uplink bandwidth resources[2] as shown in Fig. 1. Overwhelming users with excessive amount of Interests delays Data packets in respond to Interests, and consequently, the quality of the delivery suffers, *e.g.,* VoD viewers can experience buffering events due to excessive jitter. The resolution also needs to consider the overhead

of the traffic on the ISP network as well. Ideally, a load-aware resolution mechanism should distribute the content requests across users within an ISP and efficiently pool the uplink resources of the users providing content.

A related challenge is the awareness to up-to-date locations of cached content stored at the users. In a decentralised scenario with untrusted users sharing their cached content, one can not allow information on named content locations (*i.e.,* user locations) to be advertised in the control plane without restrictions. Otherwise, a user could flood the network with advertisements of contents that are not available at the user. This could result with a large amount of incorrect state to be maintained by the resolution system and Interests to be incorrectly forwarded to attackers. Ideally, the resolution system, before advertising its location, should verify that a piece of content is stored at or recently delivered to that location.

Another important challenge for the name resolution is the scalability. In particular, keeping track of the availability of chunks within the user caches for a large number of content catalogue requires significant amount of state to be maintained, even in the case of a single ISP. Also, users frequently joining and leaving can result with large number of updates to the resolution state. However, users typically download content in their entirety and a resolution mechanism can take advantage of this. Ideally, the resolution mechanism should limit forwarding state to at least content-level information (especially at the "core" routers of the ISP where majority of the traffic flows) rather than chunk-level information in order to achieve scalability.

Given the above challenges with the name resolution in a P2P content retrieval system for an ISP, we consider several possible solutions based on stateful and adaptive forwarding mechanisms of NDN. Our first solution is based on *a Satisfied Interest Table (SIT)* [39] which contains names of recently forwarded (and ideally verified) content chunks together with the interfaces where they are forwarded to. In this solution, forwarders near the end-users monitor delivery of content and notify the rest of the nodes of the successful content deliveries. Our second solution is based on a directory service under the control of the ISP such as NDNS [11].

Before we describe these solutions in detail, one final remark is on the privacy of the system. An important feature to keep in mind is to protect consumer and producer privacy in a collaborative content retrieval system. Because Interests do not contain source information, consumer information (*i.e.,* identifier and location) is largely preserved. The main challenge is to keep producer privacy.

---

[2]Netflix recommends 25 Mbps bandwidth for Ultra HD videos, which is higher than the upstream bandwidth capacity provided by majority of fiber subscriptions in the United Kingdom [10]

In our solutions, the resolution information contains a name for a region rather than the exact producer location and identity.

5.1 Adaptive Forwarding Strategy

In this solution, edge or access routers, which are directly attached to users, store the recently satisfied PIT entries in their SIT tables. These entries are periodically processed by the forwarders and used to compute forwarding state. The forwarding state contains name to interface mappings together with a *count of the number of chunks* that have been forwarded along each interface. We assume a convention of naming content chunks in the suffix of the name with numbers starting from one, and the name prefixes constitute the content names (*e.g.,* name of a video). The content names can be either IPFS names or human readable names. In the former case, an IPFS name for an NDN name can be stored at and obtained from the DHT of IPFS.

As an example, SIT entries for a content name and particular interface for first through tenth chunks are summarised in the FIB table as the name mapping to ten chunks and the corresponding interface. In case of missing chunks in the SIT, *i.e.,* user downloads a subset of chunks of a content with missing pieces, no forwarding information is added to the FIB as it is not downloaded fully; however, we expect this to be rare as users typically download video content, which is the focus of this study, from the beginning.

FIB entries of the edge routers are periodically advertised within an ISP network using an intra-domain routing protocol, and the rest of the nodes compute their forwarding entries accordingly. In the advertisements, edge routers place their own location associated with the set of content names and chunk counts rather than the users in order to maintain producer anonymity. The computed chunk count for a content name and interface is the largest of the counts associated with all the advertisements for the content received over the interface.

We assume that the edge routers perform a content integrity verification (*i.e.,* signature verification in the case of human-readable NDN names or a cheaper hash computation in the case of self-certifying IPFS names). This solution, therefore, ensures that a content is downloaded from a user before advertising its location and have network advertise their location instead of untrusted users.

We consider two load-balancing mechanisms: one based on PIT entries to infer the load on the nearby producers, and the other based on RTT measurements. In this approach, the measurement state is also used for the update of SIT entries upon producers leaving the network (*e.g.,* going offline) or replacing content with recently downloaded one in its storage. In this case, the measurements and PIT timeout events will trigger removal of entries.

A down-side of this approach is the periodic updates that need to be sent to the entire ISP network and having the core routers maintain per-content forwarding state which can incur a large memory footprint. The SIT table can be implemented as a cache with limited size containing satisfied PIT entries and can be stored as part of the in-network cache of each forwarder. Therefore, the SIT table does not incur significant additional overhead in terms of memory footprint. In this approach, the self-certifying (IPFS) naming scheme are preferable to hierarchical names for its lightweight

hash-based integrity verification for content as opposed to a more heavy-weight signature-based content integrity verification.

5.2 Directory-based Resolution

We extend the previous approach with an indirection-based one in order to achieve better scalability. In this approach, content names are registered along with a forwarding hint to a directory service such as NDNS. The registering operation can be done by the producers themselves. However, in this case we assume that users will provide a proof that they have the file stored, *i.e.,* Proof-of-Replication [18], before they are allowed to register in the directory. We borrow the Proof-of-Replication solution of IPFS project where a zero-knowledge proof is composed by the users and verified by the directory system during registry.

In order to achieve producer anonymity in a loose sense and perform load-balancing, the forwarding hints for users are designated as in-network location (*i.e.,* forwarder) names which are at most few hops away from the users. We assume that users will be able to obtain forwarding hints associated with their own locations during a bootstrapping phase by simply sending an Interest containing a special name, *e.g.,* /forwarding_hint [13], to their first-hop forwarders. The name resolution in this approach involves obtaining a forwarding hint and placing the hint in the Interest. The forwarders simply route the Interest to the location identified in the hint. From this point, the SIT state maintained at edge forwarders perform forwarding directly on the name of the content. We assume that the ISP network is divided into a small number of edge regions in which forwarders compose their forwarding state by observing satisfied PIT entries. Once an Interest reaches an edge region, the forwarders perform load-balancing to distribute the Interests among all the available producers.

6 EVALUATION

In this section, we evaluate in-network resolution and directory-based solutions in terms of their load-balancing performance using the Icarus simulator [38]. Icarus is a packet-level caching simulator for ICN architectures. We first describe the setting and then discuss the results.

6.1 Evaluation Setting

Application: We consider *VoD player* as the application running on the end users. We consider a scenario where user applications consume videos from a catalogue of size 100, where each video consists of 10K chunks. We fix the number of videos to 100 in order to limit the duration of simulations. We assume that each VoD chunk contains eight frames of a video and they are consumed by the applications at a rate of 24 frames per second (as in HD content) while playing the video[3]. We consider different playback buffer sizes in the experiments with a default of five seconds (120 frames).

Topology: We perform simulations in an ISP topology, *i.e.,* Tiscali from Rocketfuel dataset with 240 nodes. 79 of the 240 nodes have a degree of one, and we designate these nodes as access nodes and attached ten end users to each one of those nodes. Five routers

---

[3]We make certain simplifying assumptions such as no rate-adaptation at the consumers and video is transmitted in raw form as opposed to an encoded form that transmits the difference of subsequent frames.

that have the highest connectivity are designated as border routers. We assume that the ISP topology links have infinite bandwidth, while the end users have a limited upload bandwidth which can serve 20 chunks per second[4]. We assume that by default the network does not have a (in-network) cache infrastructure and relies solely on P2P content retrievals from user caches.

Workload and Storage: The simulator randomly selects 10 nodes out of 790 end-users who are not currently playing a video each second to initiate the downloading and playing of a video. Each user plays at most one video at a time and stores at most two full videos to share with others, 20K chunks. Eventually, all end-users are viewing videos simultaneously and collaboratively upload chunks to each other. The distribution of video popularity is assumed to be of Zipf type with the default exponent value chosen as 0.7. The simulations continue for one hour after a warm-up period of 15 minutes.

Content Placement and Caching: We place a stable content producer for each VoD content to a randomly chosen border router. The producers are assumed to be cloud providers with an RTT of 300-800 msecs.

6.2 Performance Metrics and Strategies

Our evaluation is based on the following metrics:

- **Average number of buffering events:** This metric measures the quality-of-experience (QoE) of the users viewing a video. It counts the average number of buffering events (*i.e.,* playback disruptions) per VoD content download. The count excludes the initial buffering event of each stream during which the video is not yet played.
- **Overhead:** This metric measures the overhead of the VoD retrievals on the ISP network in terms of the average hop counts traveled by data as a response to an interest.
- **Percentage of downloads from peers:** The ratio of VoD chunk requests that are satisfied from the peers (*i.e.,* end users) as opposed to the external producers (*e.g.,* cloud).

We present the performance of the following forwarding strategies:

- **Shortest-path:** This is the baseline strategy which simply routes Interests to the closest producer (peers) without using any measurements of the load on the producers similar to NDN's best-route forwarding strategy. In the case of multiple peers with equal distance, the strategy picks one of them using a round-robin mechanism.
- **Optimal:** This forwarding strategy schedules video chunk requests at the closest producer that can meet the playback deadline of the requested chunk. The strategy has the the knowledge of both i) instantaneous load on all producers and ii) the arrival time of already scheduled requests on each producer. This strategy avoids playback interruptions while achieving a high cache hit rate and low overhead on the network.
- **Adaptive:** In this strategy, the routing use per-Interest RTT measurements to estimate the load on each producer. The first-hop access nodes (*i.e.,* at the consumer-side) make a

routing decision and attach a forwarding hint on the Interests. The forwarding hint is the name of the access node of a producer (*i.e.,* at the producer-side). The nodes prefer closer (*i.e.,* based on the number of hops) nodes that can meet the playback deadline based on the past measurements and routing information (*i.e.,* hop distance) . Once the Interests arrive at the producer-side access node, a producer is selected (if there is more than one producers) based on the current load of the producers inferred by the amount of (unsatisfied) PIT state to each producer.

- **Directory-based:** In this strategy, consumers obtain forwarding hints for a set of producers from a directory service, *i.e.,* NDNS. During the initial buffering phase, consumers randomly select a peer for each Interest and observe the RTT resulting from the arrival time of the corresponding data. We ensure that at least one of the chosen peer is the external producer with unlimited bandwidth resources. Once the initial buffering phase is over, the consumer selects peers for each request whose RTT can meet the playback deadline based on the observed RTTs per peer. The network simply routes Interest packets to the given forwarding hints of producer-side access nodes. Then, the access node at the producer-side forwards the Interests to a producer according to its SIT table. If there are more than one producer with the requested content, then the access node randomly selects a producer.

6.3 Results

In this section, we investigate the impact of i) size of playback buffer, ii) size of in-network caching, and iii) content popularity on the performance of the shortest-path, directory-based (*e.g.,* NDNS), Adaptive and Optimal forwarding strategies.

6.3.1 Impact of playback buffer size. Fig. 4(a) shows the number of buffering events for each of the four forwarding strategies. The shortest-path strategy, which simply routes interests to the closest peer, results in the highest number of buffering events where video playback is interrupted for buffering. This is simply because the shortest-path strategy is unable to share the upload overhead evenly among the users. Even worse, it overwhelms certain producers with too many requests. The shortest-path strategy results in nearly five buffering events per video stream for the smallest buffer size of one second worth of content. Even with a buffer size of ten-seconds worth of video, the shortest-path strategy still causes excessive buffering. According to our results (not shown in the Figures) only 4% of the viewers did not experience buffering events for the shortest-path strategy with the largest buffer size. 20% of streams experienced more than one buffering interruptions throughout the streaming. On the other hand, the directory-based resolution results with significantly less buffering events where at least 82% of streams do not experience any playback interruption for the smallest buffer size. In this approach, we observed that 99% of streams observed at most one buffering event, *i.e.,* no repeated buffering event. This is because users experiencing long RTTs during pulling of VoD chunks from a peer can simply switch to a different peer. However, even for the largest buffer size, around 5% of streams experience buffering events.

---

[4]Each chunk is of size max IP packet 64KB and contains eight frames of HD video content.

(a) Buffering Events
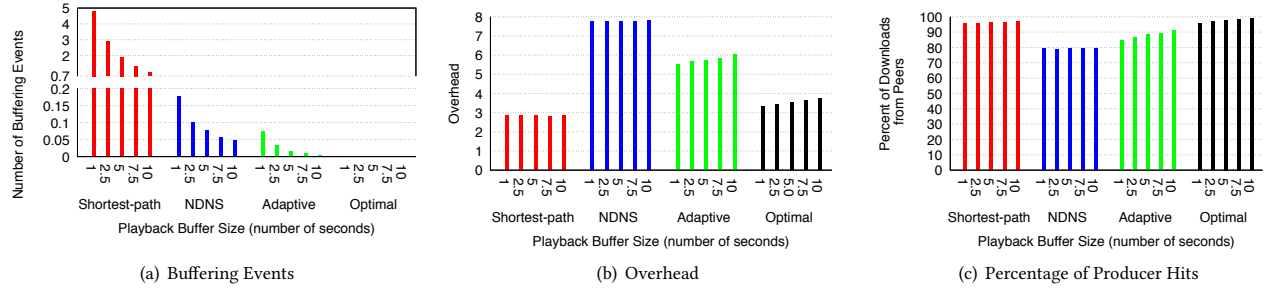
(b) Overhead

(c) Percentage of Producer Hits

Figure 4: Results with varying playback buffer size.

The adaptive streaming approach achieves a significantly (*i.e.,* over 100% improvement) better performance than the directory-based approach through its ability to detect and react to congestion at producers using i) RTT measurements at the first-hop access forwarder of the consumer and ii) load-balancing at the last-hop producer-side access node using PIT state. We observe that even for the smallest buffer size, only 7% of the streams experience buffering and less than 0.4% of users experience buffering for the largest buffer size. We observe that the optimal strategy is able to achieve no interruptions to playback for all buffer sizes through its full knowledge of the instantaneous load on the producers.

In Fig. 4(b), we observe that the directory-based strategy incurs the highest overhead in terms of average number of hops traveled by content to the network. This is because in this approach, the consumers select a peer without the knowledge of hop-distance to the peer when there is more than one peer available. Adaptive forwarding strategy incurs significantly less overhead than the directory-based approach because the first-hop node gives higher priority to closer destinations who can meet the deadline. The shortest-path approach is the other extreme where the incurred overhead is even less than optimal strategy albeit achieving the worst performance (in terms of buffering) among all the strategies. Since the optimal strategy selects the closest peer which can meet the deadline, it incurs the least possible overhead.

In terms of the percentage of video chunks downloaded from peers, we observe that the shortest-path strategy is very close to the optimal strategy as shown in Fig. 4 at the expense of achieving the worst QoE for the consumers. The adaptive and directory-based approaches achieved similar percentages of local downloads even though their overheads and buffering performances are quite different.

6.3.2 Impact of in-network caching. In the previous experiments, the forwarders did not possess any caches in the network. In this section, we examine the impact of in-network caching on the performance of the strategies in Fig. 5. In these experiments, we set the playback buffer size to the minimum size used in the previous experiments, *i.e.,* one-second worth of traffic. In the figures, the values in the x axis are the total cache budget for the network given in terms of the percentage of the total number of chunks, *i.e.,* between minimum of 0.1% to a maximum of 5% of total chunks. We uniformly divide the total cache budget on the forwarders.

In Fig. 5(a), we observe that caching significantly improve the buffering interruptions for all the strategies but the shortest-path. This is because in the shortest-path strategy, most of the popular content is available from another user nearby (*i.e.,* 2.8 hops away on average as shown in Fig 5(b)), and at the same time the users who are streaming the same content are out-of-synch (as a nature of VoD). This results with poor caching performance for most streams which flow along very short paths with small number of caches.

On the other hand, the rest of the strategies use longer paths with higher diversity than the shortest-path. Also, the consumers downloading the same content can converge on the same users as a result of measurement-based routing decisions. This leads to better caching performance. We observe in Fig 5(c) that caching can significantly reduce buffering events for the top-ten most popular content. As expected, overhead of the strategies reduce with increasing cache size as shown in Fig. 5(b).

6.3.3 Impact of content popularity. We investigate the impact of content popularity on the performance of the forwarding strategies in Fig. 6. In these experiments, we set the in-network cache size set to 0.1% of content and the playback buffer size to one second playback worth of chunks.

We observe in Fig. 6(a) that the buffering occurrences increase with the increase in popularity of content. This is because of the limited storage space at the end-users which we assume to hold at most two full videos, and we limit each user to also download three videos and immediately leave the network. As a result, interests contend for increasingly limited upload bandwidth resources as increasingly smaller number of content becomes increasingly more popular.

Consistent with the previous results, we observe that the adaptive strategy performs better than directory-based approach. Adaptive strategy is also not affected by the change in content popularity as much as the rest of the non-optimal strategies.

7 RELATED WORK

Similar to our work, a number of previous studies has considered exploiting the nearby storage of user premise devices for VoD delivery at the edges [35, 40]. Recently, P2P systems (*e.g.,* IPFS) have been evolving beyond a system of per-content swarms (*e.g.,* BitTorrent) towards achieving an ambitious goal of providing a content-centric
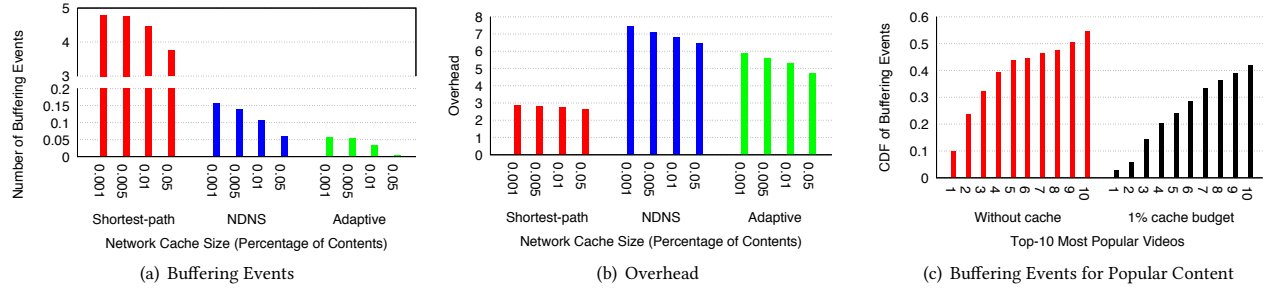
(a) Buffering Events

(b) Overhead

(c) Buffering Events for Popular Content

Figure 5: Results with varying in-network cache budget.

(a) Buffering Events

(b) Overhead
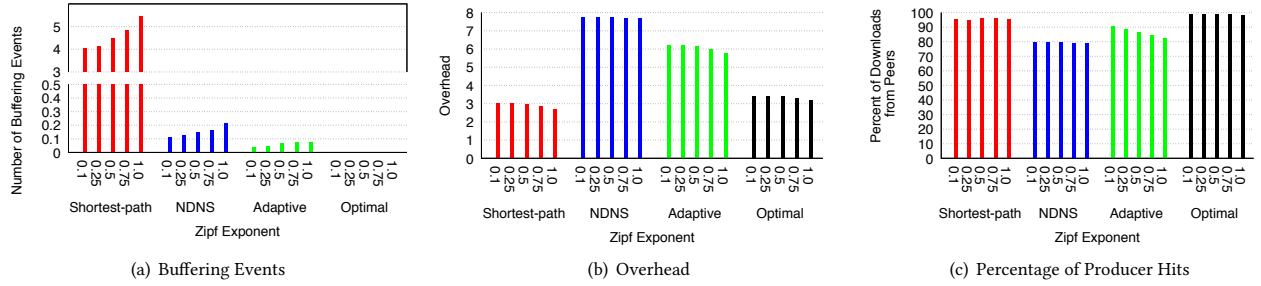
(c) Percentage of Producer Hits

Figure 6: Results with varying content popularity.

delivery service with CDN-like quality for users. Such systems consider scalability, quality-of-delivery, incentives, fairness, and trust issues, to name a few.

As pointed out by existing studies, lack of a content-centric network layer is a source of various problems for such P2P systems that work as an overlay on top of IP [31]. Mastorakis et al. has considered complementing the BitTorrent protocol with a content-centric network layer [31]. Instead, in this work, we focused on IPFS, which is a more advanced content-centric application-layer protocol suite that connects all its users in a single swarm and enables content-centric access to data. However, we have shown that IPFS has several limitations in terms of overheads and inefficiencies in data delivery which prevents IPFS to be a viable replacement for a CDN. We have proposed extending IPFS with a content-centric delivery with several extensions that are based on several existing work: secure namespace mapping [13], scalable name resolution [11, 15], stateful Interest forwarding [16, 43], and satisfied interest table (SIT) based forwarding [39].

8 CONCLUSIONS

In this work, we took a first-step towards designing a P2P content retrieval market for edge ISPs which can replace CDN content delivery within each individual ISP. We have demonstrated through ns-3 simulations using original containerised code of the IPFS that IPFS by itself is not able to take over such a task without help from the network layer in terms of load-balancing and reducing the delivery overhead for the network.

We find NDN to be a complementing network-layer architecture to IPFS. The adaptive, stateful routing mechanisms of NDN can

observe and react to congestion at the producers and steer traffic to less congested producers. Also, NDN forwarders with the addition of a Satisfied PIT Table can observe locations of content and advertise this information in the control plane.

The main take-away from this study is that adaptive load-based forwarding strategies are very useful in the network layer. However, in this study we limit the load-balancing to first- and last-hop forwarders on the path between consumers and producers. This was due to looping issues with involving too many decision making nodes on the path. As a future work, we plan to investigate more sophisticated adaptive routing techniques involving more nodes in a collaborative manner.

9 ACKNOWLEDGMENTS

REFERENCES

[1] 2018. Project CCNx. http://www.ccnx.org/.
[2] 2019. Big Buck Bunny. https://download.blender.org/peach/bigbuckbunny_movies/big_buck_bunny_480p_h264.mov
[3] 2019. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016–2021 (White Paper). (May 2019). https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-738429.html

[4] 2019. Global Internet Phenomena Report: Asia-Pacific and Europe. (May 2019). https://www.sandvine.com/press-releases/blog/sandvine-over-70-of-north-american-traffic-is-now-streaming-video-and-audio

[5] 2019. ndn-cxx: NDN C++ library with eXperimental eXtensions. Version 0.6.6. https://github.com/named-data/ndn-cxx

[6] 2019. NFD - Named Data Networking Forwarding Daemon. Version 0.6.6. https://github.com/named-data/NFD

[7] 2019. ns-3 | a discrete-event network simulator for internet systems. https://www.nsnam.org/

[8] 2019. NS3 Docker Emulator. https://chepeftw.github.io/NS3DockerEmulator/

[9] 2019. Presentation at Joint iCore and CommNet2 Workshop 2017: Content Caching and Distributed Storage for Future Communication Networks. (May 2019). https://commnet.ac.uk/wp-content/uploads/2017/06/Richard-Bradbury-The-scalability-challenge-for-broadcasting-on-the-Internet.pdf

[10] 2019. Which broadband has the fastest upload speeds? (May 2019). https://www.broadbandchoices.co.uk/ask-our-expert/which-broadband-has-the-best-upload-speeds

[11] Alexander Afanasyev, Xiaoke Jiang, Yingdi Yu, Jiewen Tan, Yumin Xia, Allison Mankin, and Lixia Zhang. 2017. NDNS: A DNS-like name service for NDN. In 2017 26th International Conference on Computer Communication and Networks (ICCCN). IEEE, 1–9.

[12] Alexander Afanasyev, Junxiao Shi, Beichuan Zhang, Lixia Zhang, Ilya Moiseenko, Yingdi Yu, Wentao Shang, Yanbiao Li, Spyridon Mastorakis, Yi Huang, Jerald Paul Abraham, Steve DiBenedetto, Chengyu Fan, Christos Papadopoulos, Davide Pesavento, Giulio Grassi, Giovanni Pau, Hang Zhang, Tian Song, Haowei Yuan, Hila Ben Abraham, Patrick Crowley, Syed Obaid Amin, Vince Lehman, and Lan Wang. 2015. NFD Developer's Guide. Technical Report NDN-0021, Revision 4. NDN. http://named-data.net/techreports.html

[13] Alexander Afanasyev, Cheng Yi, Lan Wang, Beichuan Zhang, and Lixia Zhang. 2015. SNAMP: Secure namespace mapping to scale NDN forwarding. In Computer Communications Workshops (INFOCOM WKSHPS), 2015 IEEE Conference on. IEEE.

[14] J. Arkko, B. Trammell, M. Nottingham, C. Huitema, M. Thomson, J. Tantsura, and N. ten Oever. [n. d.]. Considerations on Internet Consolidation and the Internet Architecture. Internet-Draft draft-arkko-iab-internet-consolidation-01. Internet Engineering Task Force. https://datatracker.ietf.org/doc/html/draft-arkko-iab-internet-consolidation-01 Work in Progress.

[15] Onur Ascigil, Sergi Rene, Ioannis Psaras, and George Pavlou. 2018. On-demand routing for scalable name-based forwarding. In Proceedings of the 5th ACM Conference on Information-Centric Networking. ACM, 67–76.

[16] Onur Ascigil, Vasilis Sourlas, Ioannis Psaras, and George Pavlou. 2017. A native content discovery mechanism for the information-centric networks. In Proceedings of the 4th ACM Conference on Information-Centric Networking. ACM, 145–155.

[17] Juan Benet. 2014. Ipfs-content addressed, versioned, p2p file system. arXiv preprint arXiv:1407.3561 (2014).

[18] Juan Benet, David Dalrymple, and Nicola Greco. 2018. Proof of replication. Technical Report. Technical report, Protocol Labs, July 27, 2017. https://filecoin. io/proof-of-replication. pdf. Accessed June.

[19] Weibo Chu, Lifang Wang, Haiyong Xie, Zhi-Li Zhang, and Zejun Jiang. 2016. Network delay guarantee for differentiated services in content-centric networking. Computer Communications 76 (2016), 54 – 66. https://doi.org/10.1016/j.comcom.2015.09.009

[20] Bernhard Debatin, Jennette P Lovejoy, Ann-Kathrin Horn, and Brittany N Hughes. 2009. Facebook and online privacy: Attitudes, behaviors, and unintended consequences. Journal of computer-mediated communication 15, 1 (2009), 83–108.

[21] Stefan Dziembowski, Lisa Eckey, and Sebastian Faust. 2018. FairSwap: How to fairly exchange digital goods. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. ACM, 967–984.

[22] Michael J. Freedman, Eric Freudenthal, and David Mazières. 2004. Democratizing Content Publication with Coral. In Proceedings of the 1st Conference on Symposium on Networked Systems Design and Implementation - Volume 1 (NSDI'04). USENIX Association, Berkeley, CA, USA, 18–18. http://dl.acm.org/citation.cfm?id=1251175.1251193

[23] April Glaser. 2018. How Apple and Amazon Are Aiding Chinese Censors.

[24] Anil Jangam, Prakash Suthar, and Milan Stolic. 2019. Supporting QoS aware Data Delivery in Information Centric Networks - draft-anilj-icnrg-icn-qos-00. Technical Report. Internet Engineering Task Force. https://tools.ietf.org/html/draft-anilj-icnrg-icn-qos-00 Work in Progress.

[25] Michał Król, Alberto Sonnino, Mustafa Al-Bassam, Argyrios Tasiopoulos, and Ioannis Psaras. 2019. Proof-of-Prestige A Useful Work Reward System for Unverifiable Tasks. In Proceedings of the 1st International Conference on Blockchain and Cryptocurrency. IEEE.

[26] Sailesh Kumar, Jonathan Turner, and John Williams. 2006. Advanced algorithms for fast and scalable deep packet inspection. In 2006 Symposium on Architecture For Networking And Communications Systems. IEEE, 81–92.

[27] Protocol Labs. [n. d.]. Filecoin: A Decentralized Storage Network, https://filecoin.io/filecoin.pdf. ([n. d.]).

[28] Storj Labs. 2019. Storj Whitepaper. Technical Report. Storj Labs. https://storj.io/whitepaper/

[29] MaidSafe.net. 2019. MaidSafe Whitepaper. Technical Report. MaidSafe.net. https://github.com/maidsafe/Whitepapers

[30] Sergio Marti and Hector Garcia-Molina. 2006. Taxonomy of trust: Categorizing P2P reputation systems. Computer Networks 50, 4 (2006), 472 – 484. https://doi.org/10.1016/j.comnet.2005.07.011 Management in Peer-to-Peer Systems.

[31] Spyridon Mastorakis, Alexander Afanasyev, Yingdi Yu, and Lixia Zhang. 2017. ntorrent: Peer-to-peer file sharing in named data networking. In 2017 26th International Conference on Computer Communication and Networks (ICCCN). IEEE, 1–10.

[32] Petar Maymounkov and David Mazieres. 2002. Kademlia: A peer-to-peer information system based on the xor metric. In International Workshop on Peer-to-Peer Systems. Springer, 53–65.

[33] Ilya Moiseenko. 2014. Fetching content in Named Data Networking with embedded manifests. Technical Report.

[34] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. (2008).

[35] Gianfranco Nencioni, Nishanth Sastry, Gareth Tyson, Vijay Badrinarayanan, Dmytro Karamshuk, Jigna Chandaria, Jon Crowcroft, Gianfranco Nencioni, Nishanth Sastry, Gareth Tyson, et al. 2016. SCORE: Exploiting global broadcasts to create offline personal channels for on-demand access. IEEE/ACM Transactions on Networking (TON) 24, 4 (2016), 2429–2442.

[36] Ingmar Poese, Benjamin Frank, Georgios Smaragdakis, Steve Uhlig, Anja Feldmann, and Bruce Maggs. 2012. Content-aware Traffic Engineering. (2012).

[37] I. Psaras, L. Saino, M. Arumaithurai, K. K. Ramakrishnan, and G. Pavlou. 2014. Name-based replication priorities in disaster cases. In 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). 434–439. https://doi.org/10.1109/INFCOMW.2014.6849271

[38] Lorenzo Saino, Ioannis Psaras, and George Pavlou. 2014. Icarus: a Caching Simulator for Information Centric Networking (ICN). In Proceedings of the 7th International ICST Conference on Simulation Tools and Techniques (SIMUTOOLS '14). ICST, ICST, Brussels, Belgium, 10.

[39] Vasilis Sourlas, Onur Ascigil, Ioannis Psaras, and George Pavlou. 2018. Enhancing information resilience in disruptive information-centric networks. IEEE Transactions on Network and Service Management 15, 2 (2018), 746–760.

[40] Kyoungwon Suh, Christophe Diot, Jim Kurose, Laurent Massoulie, Christoph Neumann, Don Towsley, and Matteo Varvello. 2007. Push-to-peer video-on-demand system: Design and evaluation. IEEE Journal on Selected Areas in Communications 25, 9 (2007).

[41] Jake Swearingen. 2018. http://nymag.com/selectall/2018/03/when-amazon-web-services-goes-down-so-does-a-lot-of-the-web.html.

[42] Christian Tschudin and Christopher Wood. 2016. File-like icn collection (flic). Internet Engineering Task Force, Internet-Draft draft-tschudin-icnrg-flic-00 (2016).

[43] Cheng Yi, Alexander Afanasyev, Ilya Moiseenko, Lan Wang, Beichuan Zhang, and Lixia Zhang. 2013. A case for stateful forwarding plane. Computer Communications 36, 7 (2013), 779–791.

[44] Yingdi Yu, Alexander Afanasyev, David Clark, Van Jacobson, Lixia Zhang, et al. 2015. Schematizing trust in named data networking. In Proceedings of the 2nd ACM Conference on Information-Centric Networking. ACM, 177–186.

[45] Lixia Zhang, Deborah Estrin, Jeffrey Burke, Van Jacobson, James D Thornton, Diana K Smetters, Beichuan Zhang, Gene Tsudik, Dan Massey, Christos Papadopoulos, et al. 2010. Named data networking (ndn) project. Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC (2010).