        DECoupled Application Data Enroute (DECADE) Problem Statement

Abstract

   Peer-to-peer (P2P) applications have become widely used on the
   Internet today and make up a large portion of the traffic in many
   networks. In P2P applications, one technique for reducing the
   transit and uplink P2P traffic is to introduce storage capabilities
   within the network. Traditional caches (e.g., P2P and Web caches)
   provide such storage, but they can be complex (e.g., P2P caches need
   to explicitly support individual P2P application protocols), and do
   not allow users to manage resource usage policies for content in the
   cache. This document discusses the introduction of in-network
   storage for P2P applications and shows the need for a standard
   protocol for accessing this storage.

Copyright Notice

Table of Contents

1. Introduction

   Peer-to-peer (P2P) applications, including both P2P streaming and P2P
   file-sharing applications, make up a large fraction of the traffic in
   many Internet Service Provider (ISP) networks today. One way to
   reduce bandwidth usage by P2P applications is to introduce storage
   capabilities in networks. Allowing P2P applications to store and
   retrieve data from inside networks can reduce traffic on the last-
   mile uplink, as well as on backbone and transit links.

   Existing P2P caches provide in-network storage and have been deployed
   in some networks. However, the current P2P caching architecture
   poses challenges to both P2P cache vendors and P2P application
   developers. For P2P cache vendors, it is challenging to support a
   number of continuously evolving P2P application protocols, due to
   lack of documentation, ongoing protocol changes, and rapid
   introduction of new features by P2P applications. For P2P
   application developers, closed P2P caching systems limit P2P
   applications from effectively utilizing in-network storage. In
   particular, P2P caches typically do not allow users to explicitly
   store content into in-network storage. They also do not allow
   applications to specific resource and access control policies over
   the usage of in-network storage. The challenges, if not addressed,
   may lead to reduced efficiency for P2P applications, and increased
   load on the network infrastructure.

   The challenges can be effectively addressed by using a standard, open
   protocol to access in-network storage [Data_Lockers]. P2P
   applications can store and retrieve content in the in-network
   storage, as well as control resources (e.g., bandwidth, connections)
   consumed by peers in a P2P application. As a simple example, a peer
   of a P2P application may upload to other peers through its in-network
   storage, saving its usage of last-mile uplink bandwidth.

   In this document, we distinguish between two functional components of
   the native P2P application protocol: signaling and data access.
   Signaling includes operations such as handshaking and discovering
   peer and content availability. The data access component transfers
   content from one peer to another.

   In essence, coupling of the signaling and data access makes
   in-network storage complex to support various application services.
   However, these applications have common requirements for data access,
   making it possible to develop a standard protocol.

2. Terminology and Concepts

   The following terms have special meaning in the definition of the
   in-network storage system.

   In-network storage: A service inside a network that provides storage
      and bandwidth to network applications. In-network storage may
      reduce upload/transit/backbone traffic and improve network
      application performance. The position of in-network storage is in
      the core of a network -- for example, co-located with the border
      router (network attached storage) or inside a data center.

   P2P cache (peer-to-peer cache): A kind of in-network storage that
      understands the signaling and transport of specific P2P
      application protocols. It caches the content for those specific
      P2P applications in order to serve peers and reduce traffic on
      certain links.

3. The Problems

   The emergence of P2P as a major network application (especially P2P
   file sharing and streaming) has led to substantial opportunities.
   The P2P paradigm can be utilized to design highly scalable and robust
   applications at low cost, compared to the traditional client-server
   paradigm.

   However, P2P applications also face substantial design challenges. A
   particular challenge facing P2P applications is the additional stress
   that they place on the network infrastructure. At the same time,
   lack of infrastructure support can lead to unstable P2P application
   performance, in particular during peer churns and flash crowds, when
   a large group of users begin to retrieve the content during a short
   period of time, leading to stress on bandwidth-constrained access
   uplinks. A potential way to reduce network stress and improve P2P
   application performance would be to make it possible for peers that
   are on bandwidth-constrained access to put data in a place that is
   free of bandwidth constraints and also accessible by other peers.
   These problems are now discussed in further detail.

3.1. P2P Infrastructural Stress and Inefficiency

   A particular problem of the P2P paradigm is the stress that P2P
   application traffic places on the infrastructure of ISPs. Multiple
   measurements (e.g., [ipoque_Internet_Study]) have shown that P2P
   traffic has become a major type of traffic on some networks.
   Furthermore, the inefficiency of network-agnostic peering (at the P2P
   transmission level) leads to unnecessary traversal across network
   domains or spanning the backbone of a network [RFC5693].

   Using network information alone to construct more efficient P2P
   swarms is not sufficient to reduce P2P traffic in access networks, as
   the total access upload traffic is equal to the total access download
   traffic in a traditional P2P system. On the other hand, it is
   reported that P2P traffic is becoming the dominant traffic on the
   access networks of some networks, reaching as high as 50-60% on the
   downlinks and 60-90% on the uplinks [DCIA] [ICNP] [ipoque_P2P_Survey]
   [P2P_File_Sharing]. Consequently, it becomes increasingly important
   to reduce upload access traffic, in addition to cross-domain and
   backbone traffic.

The inefficiency of P2P is also observed when traffic is sent
upstream as many times as there are remote peers interested in
getting the corresponding information. For example, the P2P
application transfer completion times remain affected by potentially
(relatively) slow upstream transmission. Similarly, the performance
of real-time P2P applications may be affected by potentially
(relatively) higher upstream latencies.

3.2. P2P Cache: A Complex Type of In-Network Storage

An effective technique to reduce P2P infrastructural stress and
inefficiency is to introduce in-network storage. A survey of
existing in-network storage systems can be found in [RFC6392].

In the current Internet, in-network storage is introduced as P2P
caches, either transparently or explicitly as a P2P peer. To provide
service to a specific P2P application, the P2P cache server must
support the specific signaling and transport protocols of the
specific P2P application. This can lead to substantial complexity
for the P2P cache vendor.

First, there are many P2P applications on the Internet (e.g.,
BitTorrent, eMule, Flashget, and Thunder for file sharing; Abacast,
Kontiki, Octoshape, PPLive, PPStream, and UUSee for P2P streaming).
Consequently, a P2P cache vendor faces the challenge of supporting a
large number of P2P application protocols, leading to product
complexity and increased development cost.

Second, a specific P2P application protocol may evolve continuously
to add new features or fix bugs. This in turn forces a P2P cache
vendor to continuously monitor application updates to track such
changes, leading to product complexity and increased costs.

Third, many P2P applications use proprietary protocols or support
end-to-end encryption. This can render P2P caches ineffective.
Therefore, these three problems make it difficult to use the P2P
cache as a network middlebox to support P2P application distribution.

Finally, an end host has better connectivity and connection quality
to a P2P cache than to a remote peer. Without the ability to manage
bandwidth usage, the P2P cache may increase the volume of download
traffic, which runs counter to the reduction of upload access
traffic.

3.3. Ineffective Integration of P2P Applications

   As P2P applications evolve, it has become increasingly clear that
   usage of in-network resources can improve the user's experience. For
   example, multiple P2P streaming systems seek additional in-network
   resources during a flash crowd, such as just before a major live
   streaming event. In asymmetric networks, when the aggregated upload
   bandwidth of a channel cannot meet the download demand, a P2P
   application may seek additional in-network resources to maintain a
   stable system.

   However, some P2P applications using in-network infrastructural
   resources require flexibility in implementing resource allocation
   policies. A major competitive advantage of many successful P2P
   systems is their substantial expertise in how to most efficiently
   utilize peer and infrastructural resources. For example, many live
   P2P systems have specific algorithms to select those peers that
   behave as stable, higher-bandwidth sources. Similarly, the higher-
   bandwidth sources frequently use algorithms to choose to which peers
   the source should send content. Developers of these systems continue
   to fine-tune these algorithms over time.

   To permit developers to evolve and fine-tune their algorithms and
   policies, the in-network storage should expose basic mechanisms and
   allow as much flexibility as possible to P2P applications. This
   conforms to the end-to-end systems principle and allows innovation
   and satisfaction of specific business goals. Existing techniques for
   in-network storage in P2P applications lack these capabilities.

4. Usage Scenarios

   Usage scenarios are presented to illustrate the problems in both
   Content Distribution Network (CDN) and P2P scenarios.

4.1. BitTorrent

   When a BitTorrent client A uploads a block to multiple peers, the
   block traverses the last-mile uplink once for each peer. After that,
   the peer B that just received the block from A also needs to upload
   through its own last-mile uplink to others when sharing this block.
   This is not an efficient use of the last-mile uplink. With an
   in-network storage server, however, the BitTorrent client may upload
   the block to its in-network storage. Peers may retrieve the block
   from the in-network storage, reducing the amount of data on the
   last-mile uplink. If supported by the in-network storage, a peer can
   also save the block in its own in-network storage while it is being
   retrieved; the block can then be uploaded from the in-network storage
   to other peers.

   As previously discussed, BitTorrent or other P2P applications
   currently cannot explicitly manage which content is placed in the
   existing P2P caches, nor can they manage access and resource control
   policies. Applications need to retain flexibility to control the
   content distribution policies and topology among peers.

4.2. Content Publisher

   Content publishers may also utilize in-network storage. For example,
   consider a P2P live streaming application. A Content Publisher
   typically maintains a small number of sources, each of which
   distributes blocks in the current play buffer to a set of P2P peers.

   Some content publishers use another hybrid content distribution
   approach incorporating both P2P and CDN modes. As an example,
   Internet TV may be implemented as a hybrid CDN/P2P application by
   distributing content from central servers via a CDN, and also
   incorporating a P2P mode amongst end hosts and set-top boxes.
   In-network storage may be beneficial to hybrid CDN/P2P applications
   as well to support P2P distribution and to enable content publisher
   standard interfaces and controls.

   However, there is no standard interface for different content
   publishers to access in-network storage. One streaming content
   publisher may need the existing in-network storage to support
   streaming signaling or another such capability, such as transcoding
   capability, bitmap information, intelligent retransmission, etc.,
   while a different content publisher may only need the in-network
   storage to distribute files. However, it is reasonable that the
   application services are only supported by content publishers'
   original servers and clients, and intelligent data plane transport
   for those content publishers are supported by in-network storage.

   A content publisher also benefits from a standard interface to access
   in-network storage servers provided by different providers. The
   standard interface must allow content publishers to retain control
   over content placed in their own in-network storage and to grant
   access and resources only to the desired end hosts and peers.

   In the hybrid CDN/P2P scenario, if only the end hosts can store
   content in the in-network storage server, the content must be
   downloaded and then uploaded over the last-mile access link before
   another peer may retrieve it from an in-network storage server.
   Thus, in this deployment scenario, it may be advantageous for a
   content publisher or CDN provider to store content in in-network
   storage servers.

5. Security Considerations

   There are several security considerations related to in-network
   storage.

5.1. Denial-of-Service Attacks

   An attacker can try to consume a large portion of the in-network
   storage, or exhaust the connections of the in-network storage through
   a denial-of-service (DoS) attack. Authentication, authorization, and
   accounting mechanisms should be considered in the cross-domain
   environment. Limitation of access from an administrative domain sets
   up barriers for content distribution.

5.2. Copyright and Legal Issues

   Copyright and other laws may prevent the distribution of certain
   content in various localities. In-network storage operators may
   adopt system-wide ingress or egress filters to implement necessary
   policies for storing or retrieving content, and applications may
   apply Digital Rights Management (DRM) to the data stored in the
   network storage. However, the specification and implementation of
   such policies (e.g., filtering and DRM) is not in scope for the
   problem this document proposes to solve.

5.3. Traffic Analysis

   If the content is stored in the provider-based in-network storage,
   there may be a risk to privacy: a malicious service provider could
   use some link that a victim user is interested in, estimate that
   another user accessing the same data may have the same interest, and
   use this information as a basis to perform a phishing attack on the
   other user.

5.4. Modification of Information

   This type of threat means that some unauthorized entity may alter
   in-transit in-network storage access messages generated on behalf of
   an authorized principal in such a way as to effect unauthorized
   management operations, including falsifying the value of an object.
   This threat may result in false data being supplied either because
   the data on a legitimate store is modified or because a bogus store
   is introduced into the network.

5.5. Masquerade

   This type of threat means that an unauthorized entity gains access to
   a system or performs a malicious act by illegitimately posing as an
   authorized entity. In the context of this specification, when
   accessing in-network storage, one malicious end host can masquerade
   as another authorized end host or application server to access a
   protected resource in the in-network storage.

5.6. Disclosure

   This type of threat involves the danger of someone eavesdropping on
   exchanges between in-network storage and application clients.
   Protecting against this threat may be required as a matter of
   application policy.

5.7. Message Stream Modification

   This type of threat means that messages may be maliciously
   re-ordered, delayed, or replayed to an extent greater than what would
   occur in a natural network system, in order to effect unauthorized
   management operations on in-network storage. If the middlebox (such
   as a Network Address Translator (NAT)) or proxy between an end host
   and in-network storage is compromised, it is easy to do a stream
   modification attack.

6. Acknowledgments

   We would like to thank the following people for contributing to this
   document:

   Ronald Bonica

   David Bryan

   Kar Ann Chew

   Lars Eggert

   Roni Even

   Adrian Farrel

   Yingjie Gu

   David Harrington

   Leif Johansson

Francois Le Faucheur

Hongqiang Liu

Tao Ma

Borje Ohlman

Akbar Rahman

Peter Saint-Andre

Robert Sparks

Sean Turner

Yu-shun Wang

Richard Woundy

Yunfei Zhang

7. Informative References

   [DCIA]   Parker, A., "P2P Media Summit presentation", Distributed
               Computing Industry Association, October 2006,
               <http://www.dcia.info/activities/p2msla2006/
               CacheLogic.ppt>.

   [Data_Lockers]
               Yang, Y., "Open Content Distribution using Data Lockers",
               CoXNet Workshop, Beijing, China, November 2010,
               <http:// cs-www.cs.yale.edu/homes/yry/projects/p4p/
               open-data-lockers-nov-2010-coxnet.pdf>.

   [ICNP]   Wu, H., "Challenges and Opportunities of Internet
               Developments in China", ICNP 2007 Keynote Speech,
               October 2007,
               <http://www.ieee-icnp.org/2007/keynote_D.html>.

   [P2P_File_Sharing]
               Casadesus-Masanell, R. and A. Hervas-Drane, "Peer-to-Peer
               File Sharing and the Market for Digital Information
               Goods", Journal of Economics & Management Strategy,
               Vol. 19, No. 2, pp. 333-373, Summer 2010.

   [RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic
              Optimization (ALTO) Problem Statement", RFC 5693,
              October 2009.

   [RFC6392] Alimi, R., Ed., Rahman, A., Ed., and Y. Yang, Ed., "A
              Survey of In-Network Storage Systems", RFC 6392,
              October 2011.

   [ipoque_Internet_Study]
              Schulze, H. and K. Mochalski, "Internet Study 2008/2009",
              2009, <http://www.ipoque.com/resources/internet-studies>.

   [ipoque_P2P_Survey]
              "ipoque's 2007 P2P Survey to be presented at Technology
              Review's Emerging Technologies Conference at MIT",
              August 2007, <http://www.ipoque.com/en/news-events/
              press-center/press-releases/2007/
              ipoque%C2%B4s-2007-p2p-survey-to-be-presented-at-
              technology>.

Authors' Addresses

   Haibin Song
   Huawei
   101 Software Avenue, Yuhua District
   Nanjing, Jiangsu Province 210012
   China

   EMail: haibin.song@huawei.com

   Ning Zong
   Huawei
   101 Software Avenue, Yuhua District
   Nanjing, Jiangsu Province 210012
   China

   EMail: zongning@huawei.com

   Y. Richard Yang
   Yale University

   EMail: yry@cs.yale.edu

   Richard Alimi
   Google

   EMail: ralimi@google.com