Integration Examples of DECADE System
draft-chen-decade-integration-example-00

Abstract

   DECADE is an in-network storage infrastructure which is under
   discussions and constructions. It can be integrated into Peer-to-
   Peer (P2P) applications to achieve more efficient content
   distributions. This document represents two detailed examples of how
   to integrate DECADE into P2P applications (live streaming and file
   sharing). Specifically, it describes mainly about: 1) a preliminary
   DECADE client API; 2) a P2P live streaming integration with DECADE;
   3) a P2P file sharing integration with DECADE; 4) tests on our DECADE
   integrarions and 5) an application performance analysis from the
   tests.

Status of this Memo

   This Internet-Draft is submitted to IETF in full conformance with the
   provisions of BCP 78 and BCP 79.

   Internet-Drafts are working documents of the Internet Engineering
   Task Force (IETF), its areas, and its working groups. Note that
   other groups may also distribute working documents as Internet-
   Drafts.

   Internet-Drafts are draft documents valid for a maximum of six months
   and may be updated, replaced, or obsoleted by other documents at any
   time. It is inappropriate to use Internet-Drafts as reference
   material or to cite them other than as "work in progress."

   The list of current Internet-Drafts can be accessed at
   http://www.ietf.org/ietf/1id-abstracts.txt.

   The list of Internet-Draft Shadow Directories can be accessed at
   http://www.ietf.org/shadow.html.

   This Internet-Draft will expire on September 15, 2011.

Copyright Notice

Table of Contents

1. **Introduction**

   DECADE is an in-network storage infrastructure under discussions and
   constructions. It can be integrated into Peer-to-Peer (P2P)
   applications to achieve more efficient content distributions.

   This draft introduces two instances of application integration with
   DECADE. In our example system, the core component includes DECADE
   server and DECADE-aware P2P clients (live streaming client and file-
   sharing client). A DECADE server runs at Linux platform and is
   designed to support interactive control and data transport for DECADE
   clients. For live streaming case, we deployed a P2P live streaming
   system called P2PLS (P2P Live Streaming). We also utilized a
   preliminary API (Application Programming Interface) set, which is
   supposed to be provided by DECADE, to enable P2PLS clients to
   leverage DECADE in their data transmission. For file-sharing case,
   we choose an open source P2P client software named Vuze which
   supports user defining plugin to extend software functions. In this
   draft, we introduce the structures of the both DECADE integration
   applications, the DECADE related control flow, the environment of
   tests on both integrations, and the system performance in the tests.

   Please note that P2PLS and Vuze in this draft only represent the
   usage case of "live streaming" and "file-sharing" out of a large
   number of P2P applications, while DECADE itself can support other
   applications. The API set of DECADE is an experimental design and
   implementation. It is not a standard and is still under development.
   Currently, DECADE in this draft is only a preliminary framwork of in-
   network for P2P. It is designed to demonstrate the pros and cons of
   in-network storage utilized by P2P applications rather than to reach
   a final solution.

2. **Concepts**

2.1. **P2P**

   Peer-to-Peer computing or networking is a distributed application
   architecture that partitions tasks or work loads between peers.
   Peers are equally privileged, equipotent participants in the
   application.

2.2. **DECADE Server**

   A DECADE server is implemented with DECADE protocols, management
   mechanism and storage strategies. It is an important element to
   provide DECADE services. In a DECADE server, we have a number of
   Data Lockers each of which is a virtual account and private storage

space for applications.

2.3. DECADE Module

DECADE module is a functional component for application clients to
utilize DECADE. This component serves as an application-specific
interface between a particular application and DECADE servers. It
can be a simple implementation of basic DECADE access APIs, or a
smart realization which integrates application-specific control
strategies with DECADE APIs.

2.4. P2P LiveStreaming Client (P2PLS Client)

P2P LiveStreaming Client (P2PLS Client) is our self-maintained
version of a native P2P live streaming application. It is one
example out of a large number of P2P applications.

2.5. DECADE Client

DECADE client is an integration of a native P2P client and a DECADE
module. It is not required to embed DECADE module into native P2P
clients, since it can also be an independent conponent running at a
remote server.

2.6. Vuze

Vuze is an open source P2P application, which uses BitTorrent
protocol for message and data exchanging. Vuze provides a set of
interfaces which support users to develop particular extensions.

2.7. DECADE Plugin

A plugin built into Vuze to implement DECADE functions including
getting/putting data from/to DECADE server and redirection

2.8. DECADE-Enabled Vuze

A Vuze client that is enabled by DECADE plugin.

2.9. Remote Controller

A controller which can control every Vuze client to start or stop
downloading tasks. It also has a function of collecting statistic
information of each Vuze client. It is a major operating platform.

3. **DECADE Client API**

   In order to simplify the DECADE integration with P2P clients, we
   provide an API set which covers the communications with DECADE
   servers and token-generation. On top of this API, a P2P client can
   develop its own application-specific control and data distribution
   policies.

   There are five basic interfaces:

   o Get_Object: to get an object from a DECADE server with an
      authorized token. The get operation can be classified into two
      categories: Local Get and Remote Get. Local Get is to get an
      object from a local DECADE server. Remote Get is to use a
      client's local DECADE server to indirectly get an object from a
      remote DECADE server. The object will be firstly passed to the
      local DECADE server, then returned to the application client

   o Put_Object: to store an object into a DECADE server with an
      authorized token. A client can either store an object into its
      local DECADE server or into other clients' DECADE servers, if it
      has an authorized token. Both of the operations are direct, for
      we don't provide indirectly storing (i.e. remote put).

   o Delete_Object: to delete an object in a DECADE server explicitly
      with an authorized token. Note that an object can also be deleted
      implicitly by setting an expired time or a specific TTL.

   o Status_Query: to query current status of an application itself,
      including listing stored objects, resource usage, etc.. Such
      information is private.

   o Generate_Token: to generate an authorized token. The token can be
      used to access an application client's local DECADE server, or
      passed to other clients to allow them to access the client's local
      DECADE server.

4. **DECADE Integration of P2P LiveStreaming Client**

   We integrate a DECADE module into a P2P live streaming application--
   P2PLS, in order that clients of this application can easily leverage
   DECADE in their data distributions.

4.1. DECADE Integration Architecture

   The architecture of the P2PLS application and DECADE integration is
   shown in Figure 1:

```
        +---------------+                     +---------------+
        |    DECADE     |                     |    DECADE     |
        |    Peer       |                     |    Peer       |
        |+-------------+|   +---------------+  |+-------------+|
        ||DECADE Module||---| DECADE Server |---||DECADE Module||
        |+------^------+|   +---------------+  |+------^------+|
        |   API |       |                     |   API |       |
        |+------v------+|   +---------------+  |+------v------+|
        ||Native Client||---|    Tracker    |---||Native Client||
        |+-------------+|   +---------------+  |+-------------+|
        +---------------+                     +---------------+
```
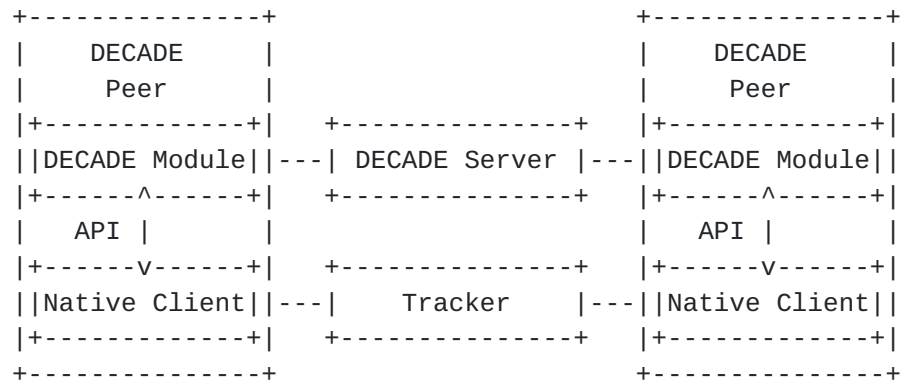
                           Figure 1

   A DECADE-integrated P2PLS client uses DECADE module to communicate
   with its DECADE server and transmit data between itself and its
   DECADE server. It is compatible with its original P2P protocol,
   while it also uses a DECADE protocol to exchange DECADE related
   messages with other peers.

4.1.1. Data Access

   DECADE module is called whenever a client wants to get data objects
   from (or put data objects into) its DECADE server. Each data object
   transferred between a client and its DECADE server should go through
   DECADE module. Neither the DECADE server and the original client
   knows each other. A data object is a data transfer unit between
   DECADE servers and application clients. Its size can be application-
   customized, according to variable requirements of performance or
   sensitive factors (e.g. low latency, high bandwidth utilization).

4.1.2. Message Control

   Control and data plane decoupling is a design principle of DECADE.
   Control messages are propagated in an original P2P way. DECADE only
   introduces an additional control message between DECADE module which
   carries DECADE authorized token. By exchanging DECADE authorized
   tokens, P2P live streaming clients can retrieve or store data objects
   into or from others' DECADE servers.

4.2. Challenges in DECADE Integration

   One essential objective of DECADE integration is to improve (or at
   least not to hurt) the application performance. However, as a brand
   new architecture, DECADE has some inherit challenges which will
   potentially be harmful to application performance. In our P2P live
   streaming case, we met mainly two such limitations of DECADE:

4.2.1. Limited Connection Slot

   Limited Connection Slot: In native P2P systems, a peer can establish
   tens or hunderds of concurrent connections with other peers.
   However, this situation can hardly be ture when it is integrated with
   DECADE because it is too expensive for DECADE servers to maintain so
   many connections for each peer. Typically, each DECADE peer only
   have m connection slots, which means it can only at most have m
   active connections with its DECADE server simultaneously. A
   potential "side effect" of limited connection slot is that the
   content availability and downloading rate might be impacted
   negatively by fewwer connections carrying data traffic. This
   requests us to adjust the peer's behavior in resource allocation,
   downloading/uploading scheduling, etc. to fully utilize the
   connection slots to archieve a satisfying and robust data
   downloading.

   o Batch Request: In order to fully utilize the connection bandwidth
      of a DECADE server and reduce overhead, a P2PLS client may combine
      multiple requests in a single request to DECADE server. Note that
      for the sake of improving data transfer efficiency in P2P live
      streaming, we may combine multiple data requests into a batch
      request. Generally, a batch may consist of different kinds of
      access requests.

   o Data Object Size: In typical P2P live streaming application, the
      size of a data block is relatively small, considering to reduce
      end-to-end transport latency. However, existing data size may
      incur large control overhead and low transport utilization. A
      larger data object size may be needed to utilize DECADE more
      efficiently.

4.2.2. Additional Control Latency

   Addtional Control Latency: In native P2P systems, when an uploader
   decides to reply a request for a piece, it sends the piece out
   directly. Nevertheless, in DECADE-aware P2P systems, the uploader
   typically only replies with a token of the piece. And then the
   downloader will leverage this token to fetch the piece from the
   uploader's DECADE server. This process obviously introduces

additional control latency compared with native P2P systems. It is
even more serious in latency sensitive applications such as P2P live
streaming. We need to consider how to reduce such additional delay
or how to compensate the lose with other inherit advantages of
DECADE.

o Range Token: One way to reduce request latency is to use range
   token. A P2PLS client may piggyback a range token when it
   propagates its bitmap to its neighbors, to indicate that all
   available pieces in the bitmap are accassable by this range token.
   Then instead of requesting specific pieces from this client and
   waiting for response, the neighbors can directly use this range
   token to access data in DECADE servers. Note that this method not
   only reduce request latency, but also reduce message overhead.

With these adjustments and strategies, DECADE clients' performance
was not impacted by the liminations of DECADE and was even better
than native clients' as we will see in following sections.

5. DECADE Integration of P2P Filesharing Client

We integrate DECADE with a popular P2P file-sharing application--
Vuze.

5.1. Vuze Client Design

Note that Vuze client is classified into two different kinds - Native
Vuze and DECADE-Enabled Vuze. When running Native Vuze, it behaves
as ordinary BitTorrent client: some Vuze clients upload data for
others to download. When using DECADE-Enabled Vuze, the
communication and data exchange processes are changed. The uploader
uploads data to its DECADE server, and downloaders download data from
DECADE servers. By this means, uplink traffic can be reduced sharply
and download performance can be improved. It is beneficial for ISPs
to save the last-mile uplink bandwidth.

5.2. DECADE-Enabled Vuze architecture Design

DECADE plugin is a key component of our demo system. It has several
interfaces with other components as following:

Interface between DECADE plugin and DECADE server: DECADE plugin can
upload and download data from DECADE server. It also includes other
functions such as user registration, application registration etc.

Interface between DECADE plugin and Vuze client: DECADE plugin can
register a listener to intercept the BitTorrent message such as

"BT_Request" message from or to Vuze client, encapsulate the data
from DECADE server into "BT_Piece" message and put the "BT_Piece"
message into incoming message queue, read the block/piece data from
the disk when seeding (to upload the data to DECADE server), and
start or stop the download tasks, all these functions are supported
in the Plugin API provided by Vuze.

Interface between DECADE plugins: When DECADE plugin intercepts the
"BT_Request" message from other Vuze clients, local DECADE plugin
sends "Redirect" message to remote DECADE plugin to authorize it to
download the piece data from the DECADE server.

Interface between DECADE plugin and Remote Controller: DECADE plugin
registers with Remote Controller after starting up, Remote Controller
can control all the Vuze clients to start, stop or resume the
download tasks through DECADE plugins.

The system architecture is as follow:

```
 _____                    _____
|  DACADE      |                  |  DECADE      |
|  Server      |----------------|  Server      |
|_____|                  |_____|
      ^                                 ^
      |                                 |
 _____|_____                    _____|_____
|      |      |                  |      |      |
| P2P  | Peer |                  | P2P  | Peer |
|      |      |                  |      |      |                    _____
|  ____v____  |                  |  ____v____  |                   | Remote       |
| |DECADE   | |                  | | DECADE  | |                   | Controller   |
| |Plugin   | |--------------  | | Plugin  | |---------------|_____|
| |_____| |                  | |_____| |                                 |
|    ^    ^   |                  |     ^      |                                 |
|    |    |__|_____|_____|_____|_____|
| PluginAPI   |                  | PluginAPI   |
|    |    |                      |    |    |
|  ____v____  |                  |  ____v____  |
| |         | |                  | |         | |
| |  Vuze   | |--------------  | |  Vuze   | |
| |_____| |                  | |_____| |
|_____|                  |_____|
```
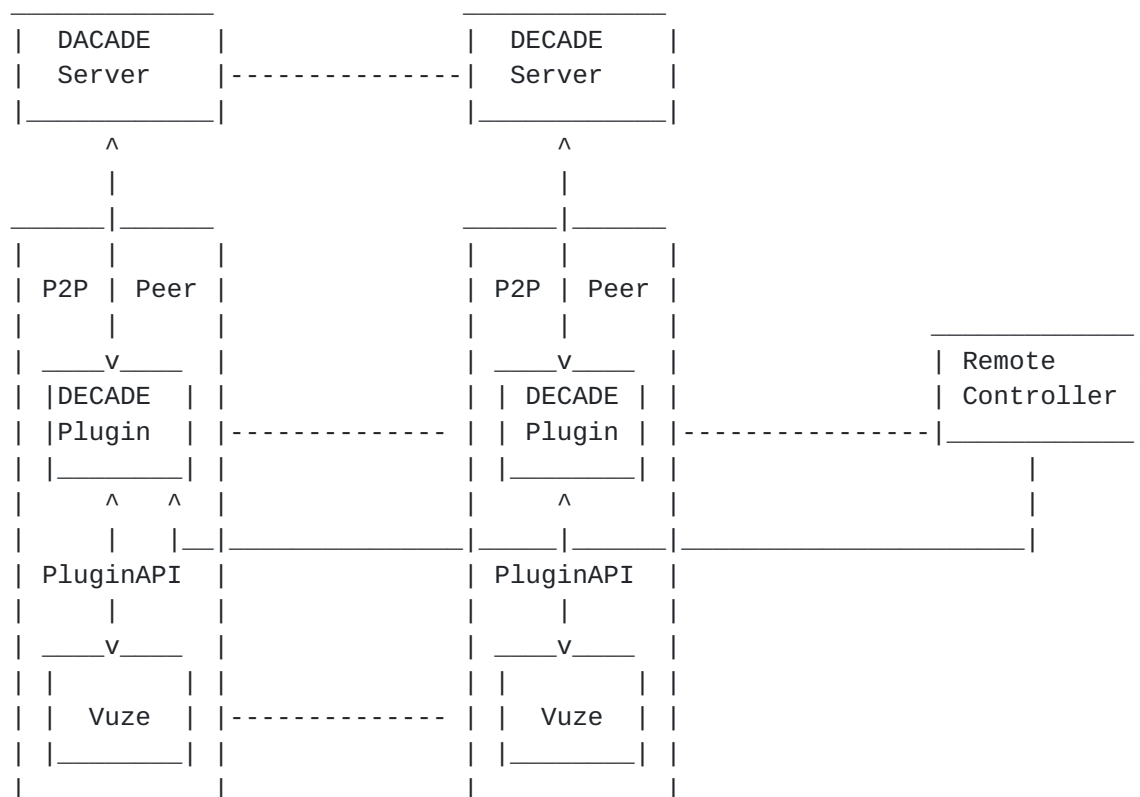

                        Figure 2

5.3. **DECADE-Enabled Vuze Communication Procedure**

   A DECADE plugin can change the data path of BitTorrent download by
   using a "Redirect" message.

   The detailed communication procedure is as following:

   o When each client starts the download task ,it will try to connect
      the tracker to get peer list and then send "BT_Request" message to
      other peers in peer list.

   o If the DECADE plugin is enabled, then it will intercept the
      incoming "BT_Request" message from other Vuze clients, and then
      reply with a "Redirect" message which includes DECADE server's
      address, authorization token and so on to the requester.

   o When a DECADE plugin receives a "Redirect" message, it will
      connect to the DECADE server according to message context and send
      "Remote Get" message to the DECADE server to request the data,
      then wait for response.

   o When a DECADE server receives a "Remote Get" message, it will
      check the server IP address in the message, Case 1: if the address
      equals to its own IP address, then it will send the data to the
      requester from its local disk/memory; Case 2: if the address is
      not equal to its own IP address, then it will send the "Remote
      Get" message to that address, to fetch the data, and then send the
      data to the requester. The data will be cached in the server
      locally for use by other requesters.

   o When a DECADE plugin obtains the data, it will encapsulate the
      data into a standard "BT_Piece" message and send to Vuze client,
      then the client can write the data block into storage.

The detailed communication diagram is as follow:

```
 _____    _____                _____    _____    _____    _____
|        |   | DACADE  |               | DACADE  |   |       |   | DECADE  |   | DECADE  |
| Vuze1  |   | Plugin1 |               | Plugin2 |   | Vuze2 |   | Server1 |   | Server2 |
|_____|   |_____|               |_____|   |_____|   |_____|   |_____|
    |             |          HandShake       |           |           |             |
    |             |                          |           |           |             |
    |k---------|------------------|---------->|           |           |             |
    |             |   Azureus HandShake|      |           |           |             |
    |k---------|------------------|---------->|           |           |             |
    |             |     BT_BitField    |      |           |           |             |
    |k---------|------------------|---------->|           |           |             |
    |             |     BT_Request    |       |           |           |             |
    |-----------|---------------->|           |           |           |             |
    |             |                          |           |           |             |
    |             |     Redirect    |        |           |           |             |
    |             |k---------------|          |           |           |             |
    |             |          DACADE RemoteGet                |           |             |
    |             |------------------------------------------>|           |             |
    |             |                   |          |        | DACADE Get |             |
    |             |                   |          |        |----------->|             |
    |             |                   |          |        | DACADE Data|             |
    |             |                   |          |        |k-----------|             |
    |             |              DACADE Data                 |           |             |
    |             |k-----------------------------------------|           |             |
    |             |                   |          |           |           |             |
    | BT_Piece |  |                   |          |           |           |             |
    |k---------|  |                   |          |           |           |             |
    |             |                   |          |           |           |             |
```
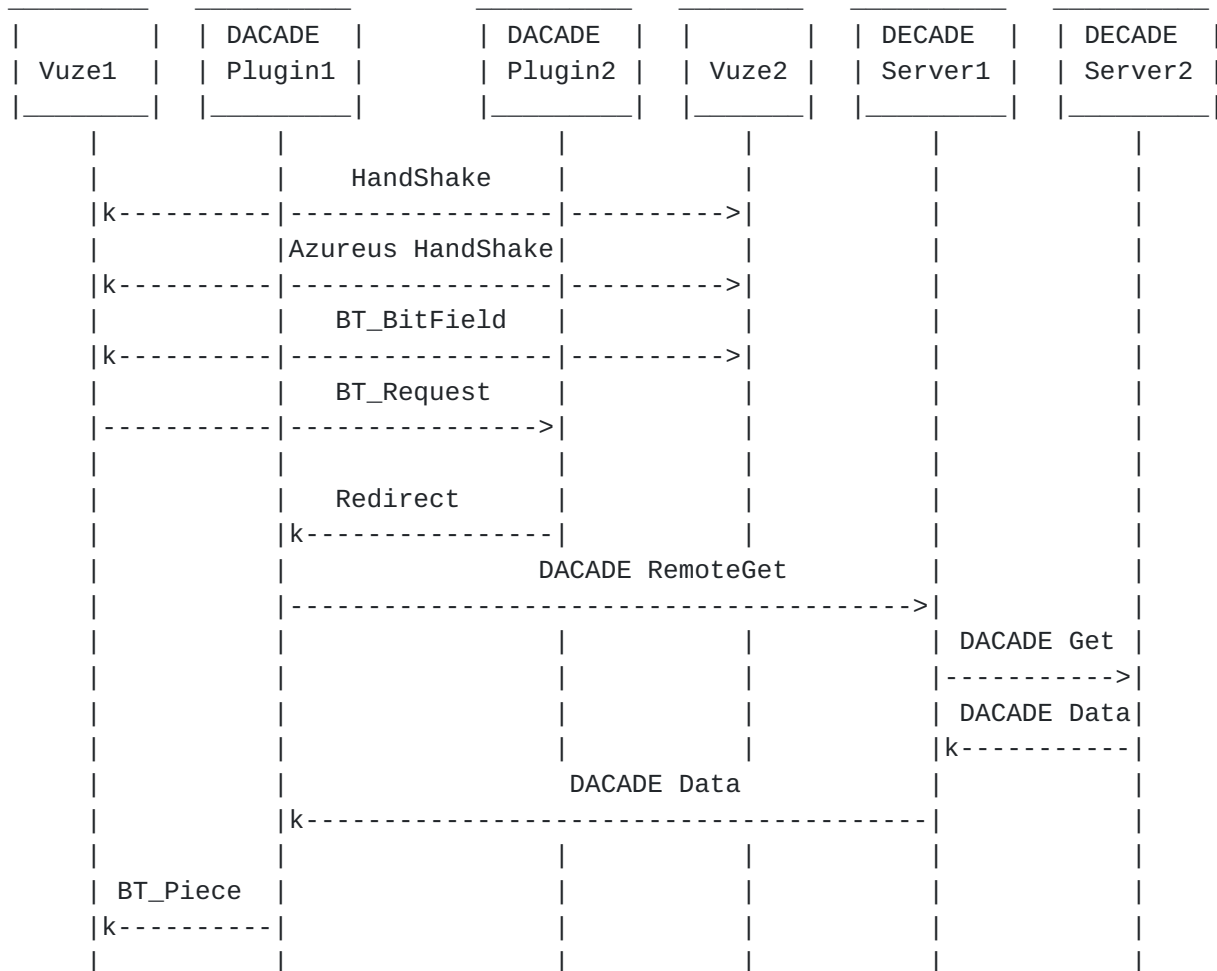
Figure 3

6. Test Environment and Settings

In order to demonstrate the performance of our DECADE implementation
and DECADE-integrated P2P live streaming and file-sharing
applications, we conduct some experimental tests in Amazon EC2 and
PlanetLab. We perform a pair of comparative experiments: DECADE
integrated P2P application v.s. native P2P application, in the same
environment using the same settings.

6.1. Test Settings

   Our tests ran on a wide-spread area and diverse platforms, including
   a famous commercial cloud platform, Amazon EC2 and a well-known
   testbed, PlanetLab. The environment settings are as following:

   o EC2 Regions: we setup DECADE servers in Amazon EC2 cloud,
      including all four regions around the world, US east, US west,
      Europe and Asia.

   o PlanetLab: we run our P2P live streaming clients and P2P file-
      sharing clients (both DECADE integrated and native clients) on
      PlanetLab of a wild-spread area.

   o Arrival pattern: we made all the clients join into the system
      within a short duration to simulate a flash crowd scenario.

   o Total Bandwidth: for a fair comparison, we set the system's total
      supply bandwidth to be exact the same in both test.

6.2. Platforms and Components of P2PLS

   In the tests, we have different functional components running in
   different platforms, including DECADE servers, P2P live streaming
   clients(DECADE integrated or Native), Tracker, Source server and Test
   Controller, as shown in Figure 4.

```
              +----------+    +------------+    +------------+
              |  P2PLS   |  /|   DECADE    |----|   DECADE   |
              | Manager  | / |   Server    |    |   Server   |
              +--------^\+ /  +-----|------+    +------|-----+ EC2
              _____\/_____|_____|_____
                       /\         |                |
              +---------/+ \  +-----|------+    +------|-----+
              |  P2PLS   ^  \ |   P2PLS     ^      ^  P2PLS   |
              | Source  |\  \|   Client    |\  /|   Client   |
              +-----|----+ \  \------^-----+ \/ +------^-----+ Planetlab
              _____|_____|_____/_____|_____
                    |         \  \   |      / \        |
              +-----|----+    v--v---v-----v   v------v-----+
              | Streaming|    |   P2PLS    |   |    Test    |
              |  Files   |    |  Tracker   |   | Controller |
              +----------+    +------------+   +------------+ Yale Lab
```
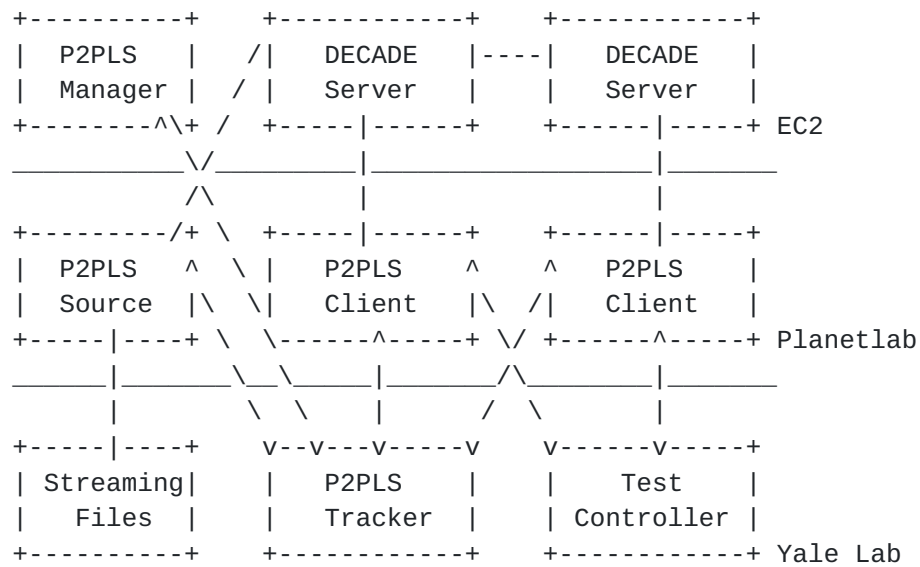
   P2PLS represents to P2P LiveStreaming.

                              Figure 4

6.2.1. EC2 DECADE Server

   DECADE Servers ran on Amazon EC2 small instances, with bandwidth
   constraint.

6.2.2. PlanetLab P2P LiveStreaming Client

   Both DECADE integrated and Native P2P live streaming clients ran on
   planetlab which spreads in various locations around the world. The
   DECADE integrated P2P live streaming clients connect to the closest
   DECADE server according to its Geo-location distance to the servers.
   DECADE integrated P2P live streaming clients use their DECADE servers
   to upload to neighbors, instead of their own "last-mile" bandwidth.

6.2.3. Tracker

   A native P2P live streaming tracker ran at Yale's laboratory and
   served both DECADE integrated clients and native clients during the
   test.

6.2.4. Source Server

   A native P2P live streaming source server ran at Yale's laboratory
   and serve both DECADE integrated clients and native clients during
   the test. The capacity of source is equivalently constrain for both
   cases.

6.2.5. Test Controller

Test Controller is a manager to control all machines' behaviors in
both EC2 and PlanetLab during the test.

6.3. Platforms and Components of Vuze

Test platforms includes Vuze client, tracker, DECADE Plugin, DECADE
Servers and other supportive components such as HTTP Server, FTP
Server and Remote Controller.

```
        +-------------+        +-------------+
        |             |        |             |
        |DECADE Server|  ...   |DECADE Server|
        |             |        |             |
        +-------------+        +-------------+
             /   \                    \
            /     \                    \
           /       \                    \
          /         \                    \
         /           \                    \
        /             \                    \
       /               \                    \
  +-------------+ +-------------+   +-------------+  +-------------+
  | Vuze client | | Vuze client |   | Vuze client |  |             |
  | with DECADE | | with DECADE | ...| with DECADE |--|   Tracker   |
  | plugin      | | plugin      |   | plugin      |  |             |
  +-------------+ +-------------+   +-------------+  +-------------+
          \             |              /
           \            |             /
            \           |            /
             \          |           /
              \         |          /
               \        |         /
                \       |        /
                 \      |       /
  +-------------+   +-------------+    +-------------+   +-------------+
  |             |   |   Remote    |    |             |   |             |
  | PL Manager  |   |  Controller |    | HTTP Server |   |  FTP Server |
  |             |   |             |    |             |   |             |
  +-------------+   +-------------+    +-------------+   +-------------+
```
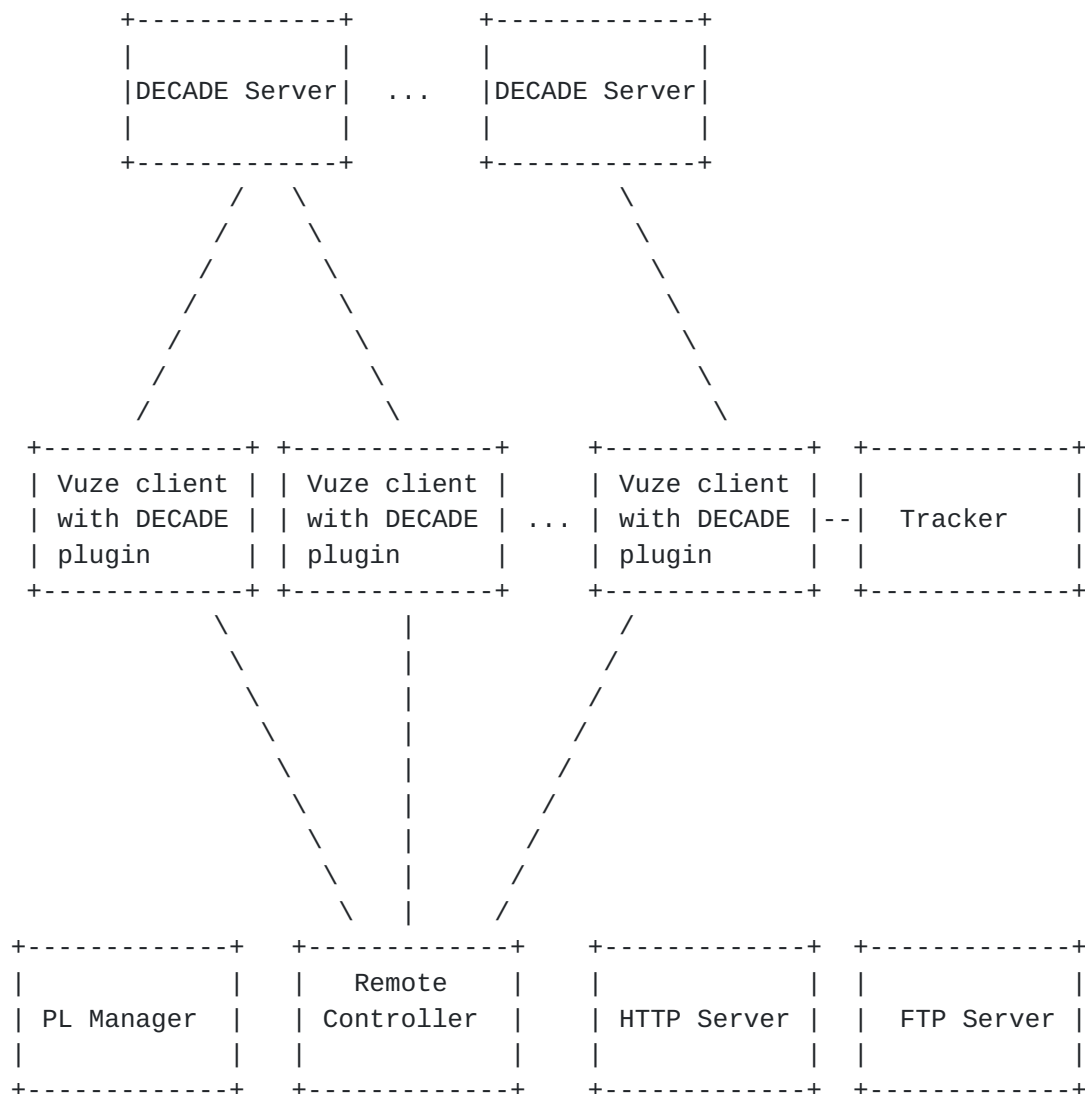
Figure 5

6.3.1. **EC2 DECADE Server**

   DECADE Servers ran on Amazon EC2 small instances, with bandwidth
   constraint.

6.3.2. **Vuze Client with DECADE Plugin**

   Vuze clients are divided into one seeding client and multiple
   leechers. Leechers run at PlanetLab, while the seeding client runs
   at the Window 2003 server. DECADE Plugin will be automatically
   loaded and run after Vuze client starts up.

6.3.3. **Remote Controller**

   Remote controller can list all the Vuze clients in user interface and
   control them to download a specific BitTorrent file. It runs at the
   same Window 2003 server with the seeding client.

6.3.4. **Tracker**

   Vuze client provides tracker capability, so we did not deploy our own
   tracker. Vuze embedded tracker is enabled when making a torrent
   file, the seeding client is also a tracker in this test.

6.3.5. **HTTP Server**

   Torrent file will be put in the HTTP Server and the leechers will
   retrieve the torrent file from HTTP Server after receiving the
   download command from Remote Controller. We use Apache Tomcat which
   is an open source software as HTTP Server.

6.3.6. **PL Manager**

   PL Manager (PlanetLab experiment Manager)is a tool developed by
   University of Washington, which presents a simple GUI to control
   PlanetLab nodes and perform common tasks such as:

   o Selecting nodes for your slice.

   o Choosing nodes for your experiment based on CoMon information
      about the nodes.

   o Reliably deploying you experiment files.

   o Executing commands / sets of commands on every node in parallel.

   o Monitoring the progress of the experiment as a whole, as well as
      viewing console output from the nodes.

7. Performance Analysis

   During the test, Both DECADE integrated P2P live streaming clients
   and DECADE integrated P2P file-sharing clients achieved impressively
   better performance than native clients.

7.1. Performance Metrics

7.1.1. P2P Live Streaming

   To measure the performance of a P2P live streaming client, we employ
   mainly four metrics:

   o Startup Delay: the duration from a peer joins the channel to the
      moment it starts to play.

   o Piece Missed Rate: the number of pieces a peer loses when playing
      over the total number of pieces.

   o Freeze Times: the number of times a peer re-buffers during
      playing.

   o Average Peer Uploading Rate: Average uploading bandwidth of a
      peer.

7.1.2. Vuze

   For the performance comparison of Native Vuze and DECADE-Enabled
   Vuze, the same system bandwidth is provided through some parameter
   settings. In Native Vuze, the system bandwidth is defined as the
   amount of the uplink bandwidth from all the Vuze clients. The
   maximum upload bandwidth (Bu) is configured to every Vuze client
   before the test. Suppose the number of the Vuze clients is N, the
   system bandwidth is Bu * N. In the DECADE-Enabled Vuze case, the same
   bandwidth (Bu * N) is configured to the corresponding Ethernet port
   of DECADE Server.

7.2. Result and Analysis

7.2.1. P2P Live Streaming

   o Startup Delay: In the test, DECADE integrated P2P live streaming
      clients startup around 35~40 seconds. some of them startup at
      about 10 seconds. Native P2P live streaming clients startup
      around 110~120 seconds. less than 20% of them startup within 100
      seconds.

   o Piece Missed Rate: In the test, both DECADE integrated P2P live
      streaming clients and native P2P live streaming clients achieved a
      good performance in pieces missed rate. Only about 0.02% of total
      pieces missed in both cases.

   o Freeze Times: In the test, native P2P live streaming clients
      suffered from more freezing than DECADE Integrated P2P live
      streaming clients by 40%.

   o Average Peer Uploading Rate: In the test, according to our
      settings, DECADE integrated P2P live streaming clients had no
      upload in their "last-mile" access network. While in the native
      P2P live streaming system, more than 70% of peers uploaded in a
      rate that is much more than streaming rate. In another word,
      DECADE can shift uploading traffic from clients' "last-mile" to
      in-network devices, which saves a lot of expensive bandwidth on
      access links..

7.2.2. **Vuze**

   Performance advantage is shown according to the test result of
   experiment:

   o

      There is no upload data traffic from Vuze clients in the DECADE-
      Enabled Vuze experiment, but in the Native Vuze experiment, the
      upload data traffic from Vuze clients is the same as download data
      traffic. Bandwidth resource is reduced in the last mile in the
      DECADE-Enabled Vuze experiment. The test result is illustrated in
      the following table. The download traffic and upload traffic
      include data traffic and signal traffic. For the upload traffic
      in the DECADE-enable Vuze, the data traffic is zero so the all
      traffic is signal overhead. Though we used the same torrent file
      in those two cases, the number of Vuze client was not the same
      because the nodes in the Planet-lab are not always stable.
      Therefore, the download traffic is a little different in two
      cases.

```
+-------------------+-------------------+-------------------+
|                   |                   |                   |
|                   | Download traffic  | Upload traffic    |
|                   |                   |                   |
+-------------------+-------------------+-------------------+
|                   |                   |                   |
| DECADE-Enabled Vuze|      480MB       |       12MB        |
|                   |                   |                   |
+-------------------+-------------------+-------------------+
|                   |                   |                   |
|    Native Vuze    |      430MB        |      430MB        |
|                   |                   |                   |
+-------------------+-------------------+-------------------+
```

                          Figure 6

   o

      Higher system resource efficiency in the DECADE-Enabled Vuze
      experiment, system resource efficiency is defined as the ratio of
      system download rate to the total system bandwidth. The test
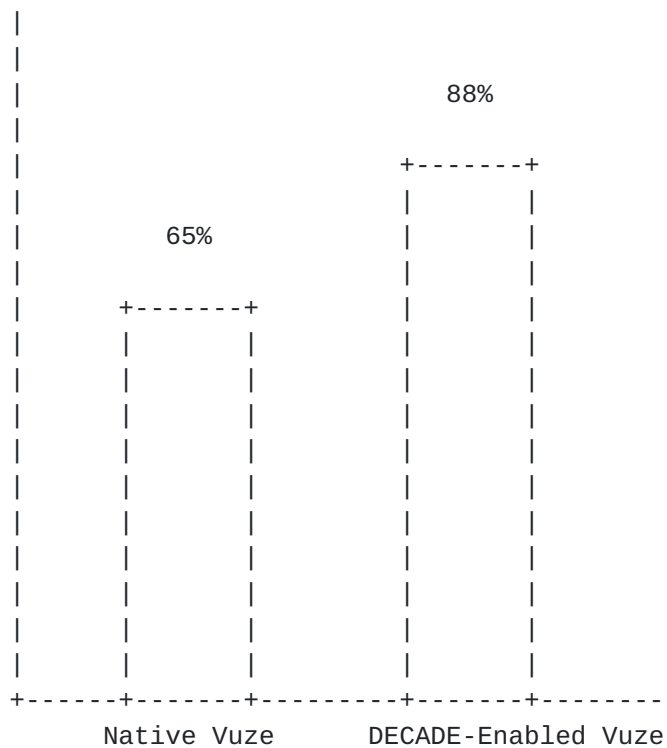   result is illustrated in the following figure.

```
     |
     |
     |                              88%
     |
     |                        +-------+
     |                        |       |
     |         65%            |       |
     |                        |       |
     |     +-------+          |       |
     |     |       |          |       |
     |     |       |          |       |
     |     |       |          |       |
     |     |       |          |       |
     |     |       |          |       |
     |     |       |          |       |
     |     |       |          |       |
     |     |       |          |       |
     |     |       |          |       |
     |     |       |          |       |
     +------+-------+---------+-------+--------
         Native Vuze       DECADE-Enabled Vuze
```

                          Figure 7

8. **Security Considerations**

   This document does not contain any security considerations.

9. **IANA Considerations**

   This document does not have any IANA considerations.

10. **Normative References**

   [RFC2119]   Bradner, S., "Key words for use in RFCs to Indicate
               Requirement Levels", BCP 14, RFC 2119, March 1997.

Authors' Addresses

   Lijiang Chen
   Yale University

   Email: lijiang.chen@yale.edu

   Hongqiang Liu
   Yale University

   Email: hongqiang.liu@yale.edu

   Zhigang Huang
   HUAWEI Technologies

   Email: hpanda@huawei.com

   Xiaohui Chen
   HUAWEI Technologies

   Email: chen.xiaohui@huawei.com