DECADE
Internet Draft
Intended status: Informational
Expires: April 2011

Xin Wang
Fudan University
Jin Zhao
Fudan University
Tiegang Zeng
Fudan University
Jun Li
Fudan University
Lei Liu
Fudan University
Shihui Duan
China CATR

October 25, 2010

Router-supported Data Regeneration for In-network Storage Systems

draft-wang-decade-data-regeneration-01.txt

Status of this Memo

Copyright Notice

Abstract

In-network storage systems store redundancy to compensate for the data loss incurred by hardware failures or other reasons. This document introduces a practical work of router-supported data regeneration in in-network storage systems to maintain the amount of redundancy. This proposed regeneration process can exploit the bandwidth diversity in the network, and the corresponding protocol enables supporting routers work transparently to support the regeneration process.

Table of Contents

1. Introduction

In-network storage systems store a substantial volume of data in an overlay network containing a large number of storage servers, which can be used for online storage service, such as file sharing, CDN, and etc. In such systems, peer churns, hardware failures and other malfunctions are unavoidable so that some data may not be accessible. Thus, the system should maintain a certain ratio of redundancy so that a subset of data is enough for data recovery.

Storing coded data of original files rather than their replicas [1] can maintain higher data integrity [2], so that any k of n storage servers can retrieve the original data. On the other hand, if a storage server fails, a replacement server should regenerate the data stored in the failed server. For coded data, this way guarantees that any k servers can retrieve the original file. The replacement server, which we call as "newcomer" in this document, should contact at least k storage servers, which we call as "providers" in this document.

The efficiency of such regeneration is influenced by the topology of the network. First, since the newcomer should contact multiple providers, several flows of data from providers may converge at some links, and thus incur a significant bottleneck in the transmission. Second, the topology may have an influence on the available bandwidth between two servers. For example, two servers in a subnet may probably have higher available bandwidth between them than two servers in two different subnets. As shown in [3], the diversity of bandwidth incurred by network topology can be exploited by letting providers relay the data flow from other providers to form a tree-structured topology during the regeneration, where network coding naturally resides, such that some slow links can be bypassed.

In this document, we show that the devices which relay the traffic during the regeneration can be not only providers, but routers as well. Routers can support the regeneration, as it can encode data when multiple data flows converge and reduce the overall traffic during the regeneration. Since the redundant maintenance is independent of data access, the scheme that we present is compatible with the protocol DECADE to access in-network storage [4]. We show that routers can support the regeneration process transparently such that no storage servers should be aware of such routers or the network topology.

2. Key problems

In order to support data regeneration in in-network storage systems, routers should be able to work transparently so that storage servers participating in the regeneration do not need any information about routers in the network. To satisfy this goal, the following key problems should be considered:

a) Bandwidth: During the regeneration, since providers are allowed to relay the traffic from other providers, available bandwidth between servers should be measured to determine the optimal routing. However, since supporting routers are supposed to be transparent to storage servers, we can only measure the end-to-end available bandwidth between each two servers participating in the regeneration process.

b) Routing: Since we can only get the table of the available bandwidth between each pair of participating servers, we first determine the routing on the overlay network covering the participating servers, then the transmission rate during the regeneration is optimized.

c) Mapping: To make the supporting routers work, we need a mapping mechanism to make supporting routes aware of the regeneration process and know how to act during the regeneration. After the routing in the overlay network has been determined, participating servers may send data to their next-hop servers, to make supporting routers between them know that there will be traffic of a regeneration process coming soon. Supporting routers should be able to determine how many flows will come, whether it is necessary to encode such flows and where the next-hop is. After mapping, the data transmission can start.

d) Reliability: Data transmission is unreliable in the network due to packet loss, disorder or other failures. Supporting routers should have a mechanism to make sure the data transmission is reliable. If the transmission between two servers is based on TCP, supporting routers should maintain the TCP state of incoming flows. If the transmission is based on UDP, there should be application-level retransmission scheme to guarantee the data reliability.

e) Congestion control: TCP provides a mechanism to control the congestion in the network. However, if multiple flows are encoded by a supporting router, the router should control the congestion in place of the destination server. However, if the transmission is based on UDP, participating servers should perform end-to-end congestion control.

3. Overview of the Router-Supported Regeneration Process

Apart from data access, data regeneration is a part of functions provided by in-network storage systems. Our scheme requires that coded data are stored in the systems, and a file is divided into k blocks and n encoded blocks are produced after encoding in which any k blocks can recover the original file. The coding technique should be decentralized and the coding operations are not necessary to perform on a single server, such as random linear coding [5]. The n encoded blocks are stored in n storage servers, i.e., each storage server stores one encoded block. When a server fails, a replacement server, called newcomer, should contact at least k encoded blocks, and reconstruct a new encoded block by re-encoding these encoded blocks.

3.1. Regeneration Process

Data regeneration process should be carried out as follows.

1. A server failure is detected and then a regeneration process is triggered. One newcomer and at least k providers are selected. A weighted complete graph covering all these k + 1 servers is made in which the weight of each edge denotes the available bandwidth measured between each pair of the k + 1 servers. A maximum spanning tree, called regeneration tree, is constructed on such a complete graph.

2. The newcomer obtains IP addresses of all providers and the regeneration tree. It then sends a NOTIFICATION messages to each provider.

3. Each provider replies an ACK message when it receives a NOTIFICATION message to its parent in the regeneration tree.

4. When an ACK message goes through a supporting router, the supporting router forwards this message and stores IP addresses of the source and the destination of the ACK message. An operation table should be constructed on the supporting router that contains the sources and destinations of the received ACK message and the number of hops to the corresponding destinations.

5. Non-leaf providers modify the type of the received ACK message to a DACK message and forward it to the newcomer. If the newcomer has received ACK or DACK message from all providers, it sends a DETECT message to each provider.

6. When a provider receives a DETECT message, it replies a RE-DETECT message to its parent in the regeneration tree. When a RE-DETECT message goes through a supporting router, the supporting router select the destination with the minimum number of hops in the operation table as the new destination of the RE-DETECT message and then forwards it. The supporting router selects IP addresses of sources of all received RE-DETECT messages and construct an encoding table.

7. Non-leaf providers modify the type of the received RE-DETECT message to a DRE-DETECT message and forward it to the newcomer. If the newcomer has received RE-DETECT or DRE-DETECT messages from all providers, it sends a START message to each provider.

8. All providers begin to send data in DATA messages that contain fixed number of bits of data to its parent node when it has received a START message. A provider that has incoming flow(s) has to wait to send its first DATA message until the first DATA message of each incoming flow has arrived and it has encoded the received data with the data it stores.

9. When a DATA message goes through a supporting router, the supporting router stores it until it has received corresponding DATA messages from all entries in its encoding table. It then encodes the data in the received DATA message and sends a new DATA message that contains the encoded data to the destination with the minimum number of hops in the operation table.

10. The newcomer stores the received data. If the newcomer has multiple incoming flows, it encodes the received data and stores them. The regeneration finishes when it has received all data.

3.2. File Regeneration Protocol

The File Regeneration Protocol (FRP) runs on the application layer, as shown in Figure 1.

```
+-----------------------------------------------------------+
| MAC    |  IP        |  TCP/UDP   | FR       |   PAYLOAD     |
| HEADER |  HEADER    |  HEADER    | HEADER   |               |
+-----------------------------------------------------------+
```

Figure 1 The overall structure of the File Regeneration Protocol

The structure of the FR head is shown as Figure 2.

```
0     1                                                          16
+------------------------------------------------------------+
|CMD |                          SOURCE                          |
|TYPE|                        IP ADDRESS                        |
+------------------------------------------------------------+
|                                                              |
|                           FILE BLOCK NAME                    |
+------------------------------------------------------------+
|RESERVE|PROVIDER  |PACKET |
|       |NUMBER    |NUMBER |
+--------------------------+
0       2          4       6
```

Figure 2 The structure of the FR header

"CMD TYPE" indicates the type of the message that includes:

NOTIFICATION: The newcomer sends a NOTIFICATION messages to providers to start a regeneration process.

ACK: A provider replies an ACK message to the newcomer when it receives a NOTIFICATION message. An ACK message makes supporting routers that it goes through be aware of the regeneration process.

DACK: A DACK message is no different from a ACK message except for the CMD TYPE. Supporting routers will not process the DACK message.

DETECT: The newcomer sends DETECT messages to providers when it has received ACK messages that contain addresses of all providers.

RE-DETECT: A provider replies an DETECT message to the newcomer when it receives a DETECT message. A RE-DETECT message makes supporting routers be aware of the number of incoming flows during the upcoming data transmission.

DRE-DETECT: A DRE-DETECT message is no different from a RE-DETECT message except for the CMD TYPE. Supporting routers will not process the DRE-DETECT message.

START: The newcomer sends START messages to all providers, indicating all servers are ready.

DATA: Providers send DATA messages that contain fixed number of bits of data to its parent in the regeneration tree. DATA messages may be encoded at supporting routers and are finally forwarded to the newcomer.

"SOURCE IP ADDRESS" represents the IP address of the last encoding device, which may be a provider or a supporting router.

"FILE BLOCK NAME" represents the name of the file block in the transmission.

"RESERVE" represents the segment that is reserved for future applications.

"PROVIDER NUMBER" represents the identifier number of the provider.

"PACKET NUMBER" represents the sequential number of the file block in the transmission.

3.3. System Implementation and Components

Router-supported data regeneration is an independent component in in-network storage systems. Since there are a large number of storage servers in the system, the server failure occurs frequently. To maintain the data integrity, a high-efficient mechanism is necessary to regenerate the lost data when a server fails. The implementation of our proposed in-network storage system contains two parts: storage servers and supporting routers. From the perspective of functions, storage servers are composed of three functional parts: dispatcher, newcomer and provider. Figure 3 illustrates the system architecture and components of the router-supported data regeneration.

```
                                   +----+   +----+
                               |---| SR |---| PR |
                               |   +----+   +----+
         +----+   +----+   +----+   |      |
         | DI |---| NC |---| SR |---|      |
         +----+   +----+   +----+   |      |
                               |   +----+   +----+
                               |---| SR |---| PR |
                                   +----+   +----+
```

Figure 3 The architecture and components of the router-supported data regeneration

DISPATCHER (DI): It manages the whole system, including the selection of the newcomer and providers and the detection of server failures.

NEWCOMER (NC): It starts the regeneration process with providers and accepts data from providers. It encodes the received data and stores the encoded data as a regenerated block.

PROVIDER (PR): Providers are storage server that provider data to the newcomer in the regeneration process.

SUPPORTING ROUTER (SR): Supporting routers are routers with computing and cache capabilities and can support regeneration. Before the data transmission during the regeneration, it collects information from ACK and RE-DETECT message to be aware of the incoming flows and the destination in the regeneration process.

3.4. Key Technologies

Some key technologies are presented in this section, which can make data transmission rate improved, the bandwidth consumption saved and the spent time reduced during the regeneration.

1. When the newcomer and providers have been selected, we measure the available bandwidth between each pair of the newcomer and providers. A complete graph covering the newcomer and providers can be constructed and the weight of each edge is the corresponding available bandwidth between two servers. A maximum spanning tree, i.e., a regeneration tree, then is constructed on this graph, in which the newcomer is the root. All non-root servers in the regeneration tree send their data to its parent. When a flow of data transferred goes through a supporting router, it may be encoded with other flows and forwarded to another server. Compared with conventional regeneration process, the method we propose utilizes the link with higher available bandwidth in the network, reduces the communication cost and thus increases the transmission rate during the regeneration.

2. Another key technology in the router-supported data regeneration is data encoding on the supporting router. During the regeneration, supporting router detects File Regeneration Protocol (FRP) by processing all IP packets that go through it. Supporting routers recognizes the file block being regenerated by analyzing the FRP. If multiple flows of the same block come during the regeneration, a supporting router should encode the received data. The header of FRP enables the supporting router to know whether encoding operations should be performed. Utilizing the computing capability, encoding operations that should have been performed on the newcomer or providers are partially transferred to supporting routers, such that supporting routers sends out only one data flow even if it receives multiple data flows. Therefore, multiple data flows sharing the same physical link are eliminated or at least partially eliminated, and transmission rate during the regeneration can be significantly improved.

4. Validation

We present our evaluation results of our proposed scheme here. We implement supporting routers on servers running Linux (kernel 2.6.30). In the network, there are four supporting routers and four storage servers, among which one is selected as both the dispatcher and the newcomer and others are providers. The storage servers and supporting routers can connect in a topology shown in Figure 4.

```
+-----+    +-----+    +-----+    +-----+
|DI+NC|----| SR1 |----| SR4 |----| PR3 |
+-----+    +-----+    +-----+    +-----+
              |          |
              |          |
              |          |
+-----+    +-----+    +-----+    +-----+
| PR1 |----| SR2 |----| SR3 |----| PR2 |
+-----+    +-----+    +-----+    +-----+
```

Figure 4 the network topology in the experiment

Each link in the network topology refers to a fast Ethernet (100BASE-TX, specifically). Actually we control the available bandwidth on each link as follows.

Table 1 The available bandwidth in the network topology

Node 1	Node 2	Available Bandwidth (MBps/S)
DI+NC	SR1	60
SR1	SR4	50
SR4	PR3	80
SR1	SR2	40
SR1	SR3	25
SR4	SR3	20
PR1	SR2	80
SR2	SR3	50
SR3	PR2	80

We regenerate a coded block with a size of 6,000,000 bytes. We compare the time spent in the regeneration process and the bandwidth consumed between the conventional regeneration process in which the newcomer receives data directly from providers and the regeneration process we propose above. The experiment is repeated for 100 times and the result is the average value.

Table 2 The average bandwidth consumption

conventional regeneration	router-supported regeneration
58241047 byte	45606648 bytes

Table 2 shows the bandwidth consumption on average. We count the total number of bytes all providers and supporting routers send out. We can see that router-supported regeneration process is able to reduce the bandwidth consumption by 21.7%, since supporting routers can encode data from multiple devices.

Table 3 The average regeneration time

conventional regeneration	router-supported regeneration
95.4 sec.	49.4 sec.

Table 3 shows the average regeneration time. Router-supported regeneration can reduce the regeneration time by 48.3%, because it can not only reduce the bandwidth consumption, but also utilize the network topology by bypassing links with low available bandwidth.

5. DECADE Compatibility

Since in the in-network storage system, servers are not guaranteed to be stable, it is necessary to maintain the data integrity by regenerating the lost block after server failures. Thus, the File Regeneration Protocol (FRP) can work as a part of DECADE protocol. According to the reliability level of the applications, DECADE-compatible applications can implement FRP independently, which will not interfere with other part of the DECADE protocol.

6. Security Considerations

This draft does not introduce any new security issues.

7. IANA Considerations

This memo includes no request to IANA.

8. Conclusions

We propose a topology-aware regeneration process for in-network storage system such that bandwidth diversity in the network can be exploited and routers may support the regeneration process by encoding the incoming data flows. We present the corresponding protocol to configure the regeneration process adaptively in which supporting routers and servers do not need to know the network topology and make decisions by their local information. System architecture is presented and related key technologies are discussed.

9. References

9.1. Normative References

[1] S. Shepler, B. Callaghan, D. Robinson, R. Thurlow, C. Beame, M. Eisler, and D. Noveck, "Network File System (NFS) version 4 Protocol", RFC 3510, 2003.

9.2. Informative References

[2] H. Song, N. Zong, Y. Yang, and R. Alimi, "DECoupled Application Data Enroute (DECADE) Problem Statement," http:// http://tools.ietf.org/id/draft-ietf-decade-problem-statement-00.txt

[3] H. Weatherspoon and J. Kubiatowicz, "Erasure Coding vs. Replication: A Quantitative Comparison," Peer-to-Peer Systems, vol. 2429/2002, pp. 328-337, 2002.

[4] J. Li, S. Yang, X. Wang, and B. Li, "Tree-structured Data Regeneration in Distributed Storage Systems with Regenerating Codes," in Proc. IEEE INFOCOM, 2010.

[5] T. Ho, R. Koetter, M. Medard, D. Karger, and M. Effros, "The Benefits of Coding over Routing in a Randomized Setting," in Proc. International Symp. Inform. Theory, pp. 442, 2003.

Authors' Addresses

Xin Wang
Fudan University
Shanghai 201203, China
Phone: 86-21-51355526
Email: xinw@fudan.edu.cn

Jin Zhao
Fudan University
Shanghai 201203, China
Phone: 86-21-51355526
Email: jzhao@fudan.edu.cn

Tiegang Zeng
Fudan University
Shanghai 201203, China
Phone: 86-21-51355526
Email: 09210240087@fudan.edu.cn

Jun Li
Fudan University
Shanghai 201203, China
Phone: 86-21-51355526
Email: 0572222@fudan.edu.cn

Lei Liu
Fudan University
Shanghai 201203, China
Phone: 86-21-51355526
Email: 09210240117@fudan.edu.cn

Shihui Duan
CATR
Beijing 100045, China
Phone: 86-10-63200068
Email: duanshihui@catr.cn