PREDICTION BASED CLOUD BANDWIDTH AND COST REDUCTION SYSTEM: PACK

A PROJECT REPORT

Presented to the Department of Electrical Engineering

California State University, Long Beach

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Electrical Engineering

Committee Members:

Anastasios Chassiakos, Ph.D. (Chair)
Wajdi Aghnatios, Ph.D.
James Ary, Ph.D.

College Designee:

Antonella Sciortino, Ph.D.

By Shubha Koundinyan

B.E, 2009, Visveswaraya Technological University

May 2016

ABSTRACT

PREDICTION BASED CLOUD BANDWIDTH AND COST REDUCTION SYSTEM: PACK

By

Shubha Koundiyan

May 2016

Predictive Acknowledgements (PACK) is the latest cloud-based end-to-end Traffic Redundancy Elimination system (TRE). Cloud-based systems have to optimize the usage of resources such as bandwidth. The bandwidth usage is reduced by decreasing the cost of TRE computation and storage. PACK is a receiver based TRE system, which uses the newly received data chunks to identify the previously received data chunks. PACK protocol diminishes the cost induced by the TRE algorithm. The server status does not have to be continuously maintained when using the PACK protocol. This feature makes PACK suitable for client mobility and server migration, which creates universal calculation environments responsible for maintaining cloud elasticity. This report presents the design and implementation of PACK protocol. Simulation results show that this design reduces the overall operational cloud cost and minimizes the application latency between the sender and the receiver.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisors Dr. Anastasios Chassiakos and Dr. Wajdi Aghnatios for their expert guidance and continuous encouragement. Without their support it would not have been possible for me to complete this project.

I would also like to extend my sincere thanks to the staff and faculty members of the Electrical Engineering Department, who have encouraged me throughout the course of my Master's Degree. I express my sincere gratitude to everyone who have directly or indirectly contributed to the successful completion of this project.

Lastly, I want to thank my family who have been a constant source of inspiration throughout my life.

TABLE OF CONTENTS

ABSTRACT.....ii

ACKNOWLEDGEMENTS……..…..……...…...……..iii

LIST OF FIGURES..……..……..….vi

LIST OF ABBREVIATIONS……..vii

1.INTRODUCTION……...……..…….……1

2.LITERATURE REVIEW…….…….....3

3.PROPOSED SYSTEM…….…….…….…….....6

4.OPTIMIZATION AND FUTURE ADVANCEMENTS…….…….17

5.RESULTS AND DISCUSSIONS…….....……..……..19

6.CONCLUSION……..……22

REFERENCES……..……...24

LIST OF FIGURES

1. Class diagram 7

2. Sequence diagram 8

3. Activity diagram 9

4. Collaborative diagram 10

5. Deployment diagram 10

6. Component diagram 11

7. Prediction queue filling 13

8. Process of analysis of prediction queues 14

9. Messages exchanged in a wire 15

10. Client window 19

11. File split on cloud 20

12. Window matching action on cloud 21

13. Traffic volume and detected redundancy 21

LIST OF ABBREVIATIONS

ACK	Acknowledgement
CDN	Content distribution network
GUI	Graphical User Interface
ISP	Internet service provider
LRU	Least recently used
OS	Operating System
PACK	Predictive Acknowledgements
PRED-ACK	Prediction Acknowledgement
RE	Redundancy elimination
SHA-1	Secure Hash Algorithm 1
SQL	Structured Query language
TRE	Traffic redundancy elimination
TCP	Transmission Control protocol
UML	Unified Modeling Language

CHAPTER 1

INTRODUCTION

Cloud Network

Cloud computing is a technology where resources and information are shared between devices on demand. Cloud enables a universal and on demand access to a pool of resources. This technology improves the efficiency and scalability of the network by reducing the overall cost.

Traffic Redundancy Elimination Techniques

A cloud customer pays only for the resources (bandwidth and storage) that they will use. This method gives the cloud environment a usage-based pricing model also termed as a pay-as-you-go service model. Data transfer costs, which directly corresponds to bandwidth saving, is the primary concern in cloud computing. Thus, to ensure judicious usage of bandwidth various Traffic Redundancy Elimination (TRE) algorithms are considered. These TRE algorithms aim at eliminating redundant data content, which reduces the bandwidth usage and thus the network cost.

The existing TRE techniques compare the sender and receiver signatures of data chunks that are parsed as per their data content prior to their transmission. The sender replaces the detected redundant chunks by its strong signature. These commercial TREs are used in enterprise networks and involve deployment of state synchronized middleboxes, which require the deployment of two or more propriety protocols. These middleboxes are deployed at the entry points of both the data center and the branch offices, therefore eliminating the repetitive and redundant traffic between them. Enterprise networks chiefly focus on reducing customer bandwidth bills. This is not beneficial to cloud service providers. Also, a rise in the dynamic

work environment requires the deployment of a client side and a server side middle box for a smooth and efficient operation. It has been seen that cloud side elasticity needs distribution of work among servers and data centers. It can be safely established that an end-to-end software based universal TRE is ideal for the cloud network.

The proposed TRE will use a standardized protocol stack which makes secured traffic possible. The existing TRE solutions are sender based, meaning a TRE applied to a cloud server has to continuously maintain the status of the client. Thus, there is a requirement of a new TRE system to satisfy the elasticity feature of a cloud network. Elasticity enables the system to dynamically adapt to the workload over time. First, cloud load balancing and power optimization require a full synchronization between server and the client. If this synchronization is not achieved, there is a loss in overall efficiency. Second, the popularity of Content Distribution Networks (CDN), which allow the service points for fixed and mobile users to dynamically change as per the location and loads, consumes a high amount of bandwidth. Furthermore, employing an end-to-end solution involves an additional computational and storage cost at the cloud side. This cost is weighed against the bandwidth saving gains. Thus, a TRE solution which emphasizes on cloud side might be less cost efficient. Also, it has been determined experimentally that a sender-based TRE solution adds an additional load on the servers.

This project involves the design and implementation of a prediction based cloud TRE system. This system uses previously received data chunks to predict the future data chunks. This designed system is receiver based, which makes it efficient.

CHAPTER 2

LITERATURE REVIEW

This project involves the design of a cloud-based TRE called Predictive Acknowledgements (PACK). PACK reduces the processing cost by offloading the cloud-based TRE server. PACK doesn't require the server to continuously monitor the clients' status. Thus making PACK suitable for pervasive computational environments that combine client mobility and server migration in order to maintain cloud elasticity. PACK uses the recently received data chunks to find the previously received chunks. PACK implementation is presented in this report.

Cloud computing compatible TREs have to deal with a large data exchange between the user and the cloud. This makes the traditional middleboxes' protocol independent reducing techniques inadequate. In this project a receiver based cloud compatible TRE, which addresses user mobility and cloud elasticity issues is developed. The PACK server doesn't have to continuously monitor and maintain the clients' status thus enabling cloud elasticity. Also, the redundancy of content arriving from multiple servers is eliminated without the use of a three-way handshake. An additional effort is imposed on the sender only when redundancy is exploited. This leads to an overall reduction of the cloud cost.

Method and Apparatus for Reducing Traffic Over Low Bandwidth Links

The method used for reducing the transmitted traffic is explained here. In a communication network, the sender identifies the data chunk to be transmitted. The digital signature of this chunk is then calculated to check whether this data chunk has been transmitted. This information is found by looking in a sender index table. The sender index table assigns unique index values to previously transmitted data chunks. If a data chunk has been transmitted before, then its unique index is included in the message that is being transmitted. The receiver

contains the index table. This index table associates the unique location of the data chunk at the receiver cache. If a previously transmitted data chunk is retransmitted, then that data chunk is retrieved based on its index value.

Redundancy in Network Traffic: Findings and Implications

Protocol independent reducing elimination techniques, commonly called middlebox, have gained popularity in the recent years due to their ability to prevent large scale duplicacy of common data on the internet. These techniques can remove redundant duplicate data strings from arbitrary network flows. There are many vendors who offer these middlebox solutions in order to improve the effective bandwidth of enterprises, datacenter and ISP links. Middleboxes focus on reducing the customer bandwidth bill. However, since cloud is a pay-as-you-go model, the customer only pays for the bandwidth he/she uses, thus making middleboxes inefficient in a cloud environment.

The existing algorithms use application of independent redundancy elimination (RE) for identifying the repetitive content from network data transfers. This approach is responsible for effectively increasing the performance of the enterprise access links. But extending this single point RE onto the network will require modification of the routing policies. This modification will be difficult and thus this technique cannot be deployed on the network level. A network-wide RE increases the overall network capacity thus reducing latencies in turn increasing the network throughput. This project involves development and implementation of SmartRE. SmartRE is designed on high-level design principles which allows it to be deployed throughout the network. This technique can also be extended to multihop networks and datacenter.

Most of the existing RE techniques work on the application level. There are also a few of them that work on individual packets. All the existing systems apply localized settings. In this

report we present a system that operates on individual packets. This system can be employed universally on all routers. This deployment will result in significantly reducing the link loads.

Disadvantages of the Existing System

1. The main objective of the existing TRE systems is to reduce the customer bandwidth bills. This objective fails for a cloud network as the user only pays for the bandwidth he/she uses.
2. Employees are no longer bounded to their office work stations due to an increase in on demand work spaces and work from home opportunities. In these scenarios deployment of a client and a server side middle-box is inefficient.
3. The existing end-to-end TRE systems fail to maintain end-to-end synchronization which might result in efficiency degradation.
4. For a cloud network, a server side TRE is needed to achieve load balancing and power optimization. For this TRE to be efficient a complete synchronization should be achieved between the server and the client. The efficiency of the system depends on its ability to obtain this synchronization. The existing systems find this hard to achieve.

CHAPTER 3

PROPOSED SYSTEM

This project report presents the design of a receiver based end-to-end TRE system which makes use of predictions for elimination of redundant traffic. In this system the incoming data chunk in the message is observed and is matched with a previously received data chunk at the receiver. With the help of the locally kept information regarding the chunk's metadata the server predictions are sent by the receiver. The server predictions are made up of the chunk's digital signature and the easy-to-identify hints for identifying the sender's future data. On the receiver side a new fingerprinting scheme called PACK is proposed. This scheme can be used instead of the traditionally used Rabin Fingerprinting. Rabin fingerprinting uses irreducible polynomials over a finite field to generate a digital signature.

Advantages of the Proposed System

1. PACK can reach processing speeds over 3 Gbps, which is 20% faster than Rabin fingerprinting.
2. Mobility related problems in quasi-mobile desktops/laptops are addressed by designing this receiver-based TRE.
3. It has been observed that there is a significant reduction in the traffic redundancy without the reduction in the computational effort. This results in the overall reduction in the cost.
4. This system implementation uses the TCP Options field and thus supports web, video steaming and all other applications.

Hardware and Software Used

1. User Interface: This system uses a Java graphical user interface

2. Hardware Interface: Java capabilities are used for the user-console interaction
3. Software: Oracle Java 1.6 is used for programming.
4. Operating Systems (OS): This designed system is compatible with Microsoft Windows and Linux based Operating systems.

Design

A class diagram is an important tool in the design of a system. Class diagram gives both the general and the detailed modeling of the project requirements. The class diagram of the designed system is given in Figure 1.

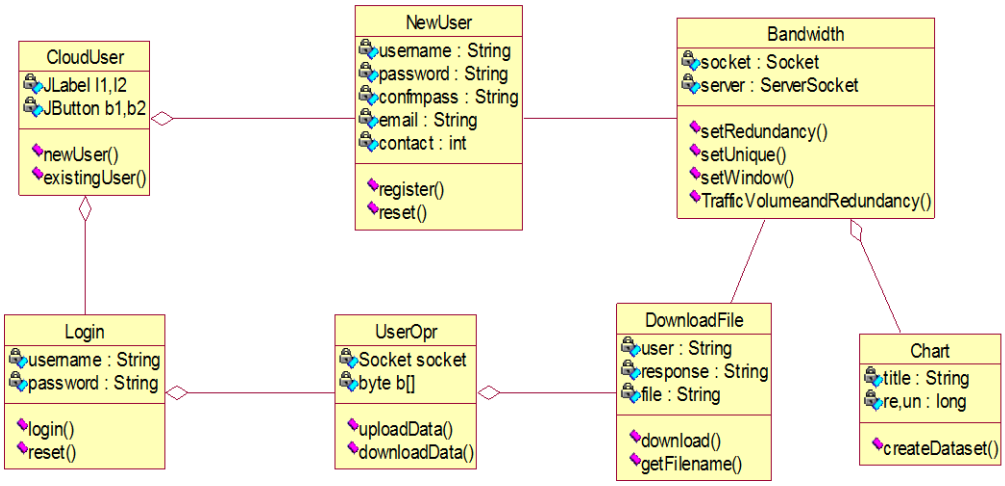

FIGURE 1. Class diagram.

Sequence diagram of a system represents how different processes operate with each other. It consists of the sequence messages exchanged between the objects to carry out the operation. The sequence diagram of the designed system is given in Figure 2.

The dynamic aspects of the system are described with the help of the activity diagram. This diagram shows the flow of control from one activity to another. The control flow can be concurrent, sequential or branched. The activity diagram of this system is given in Figure 3.

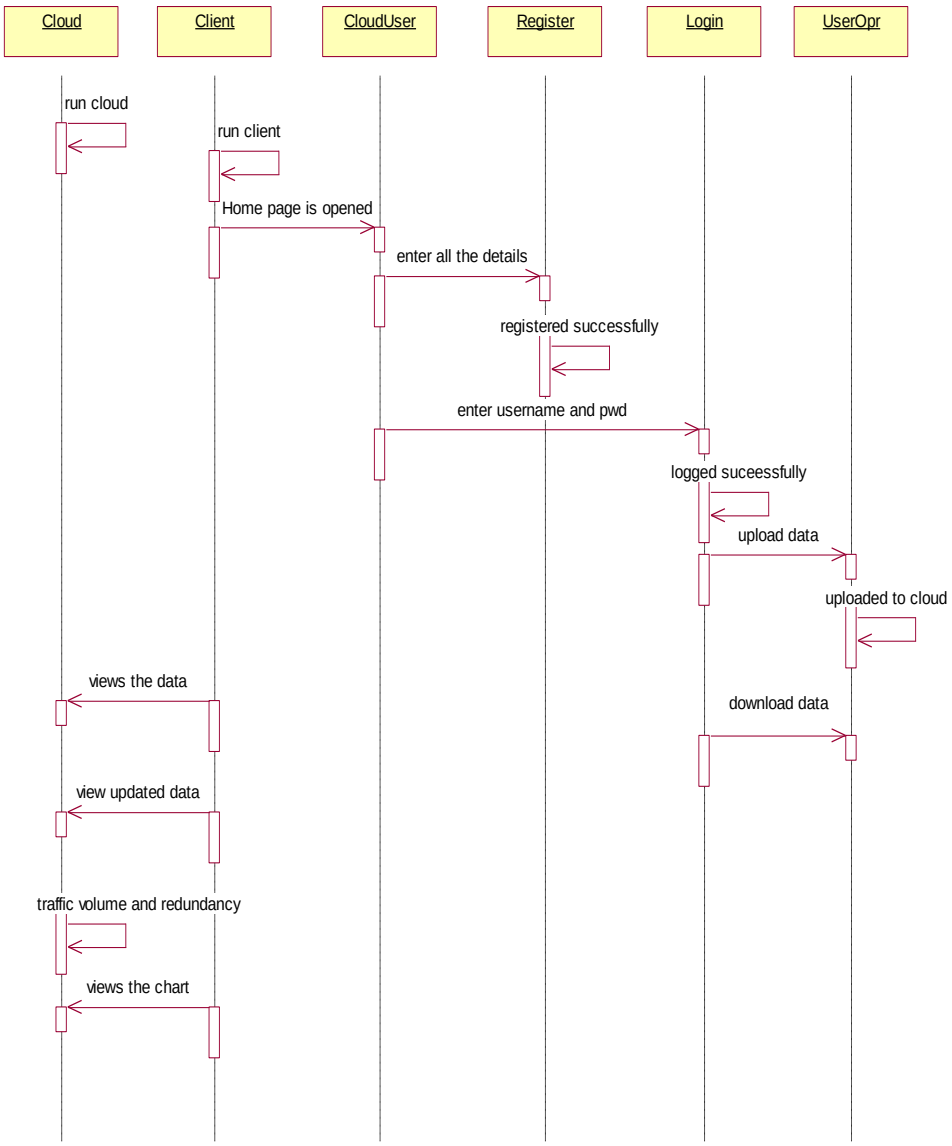

FIGURE 2. Sequence diagram.

The sequential messages that are exchanged between the different objects are given by a collaborative diagram. This diagram gives the behavior, role and functionality of each individual object in the project. The collaborative diagram for this project is given in Figure 4.

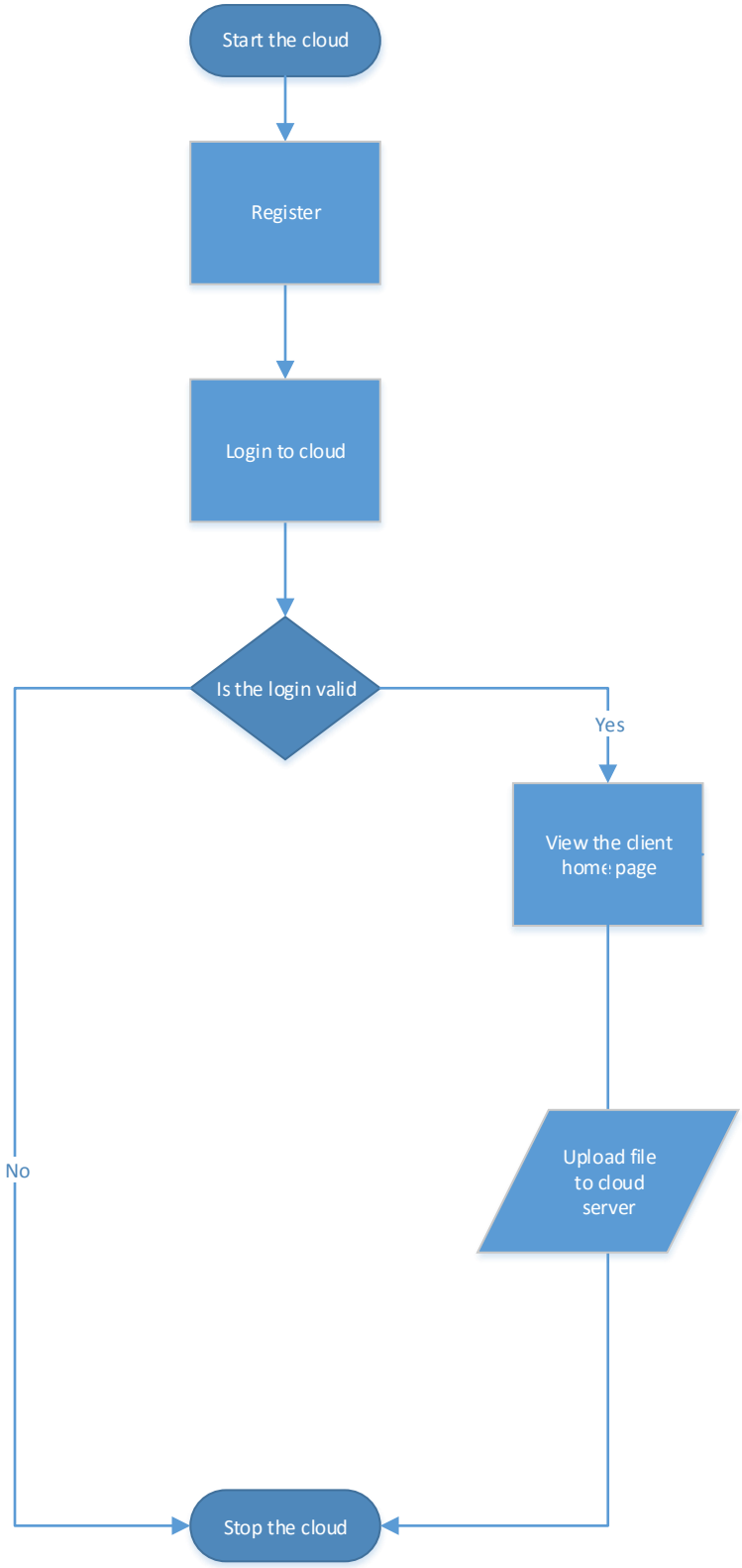

FIGURE 3. Activity diagram.

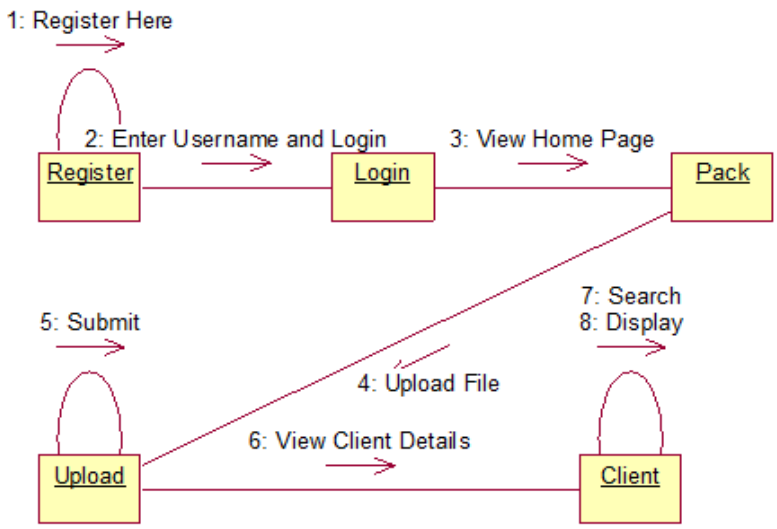

FIGURE 4. Collaborative diagram.

The different modules that are created in a project are shown by the deployment diagram in UML. Each module designed in this project is represented by a rectangle. The deployment diagram for this project is given in Figure 5.

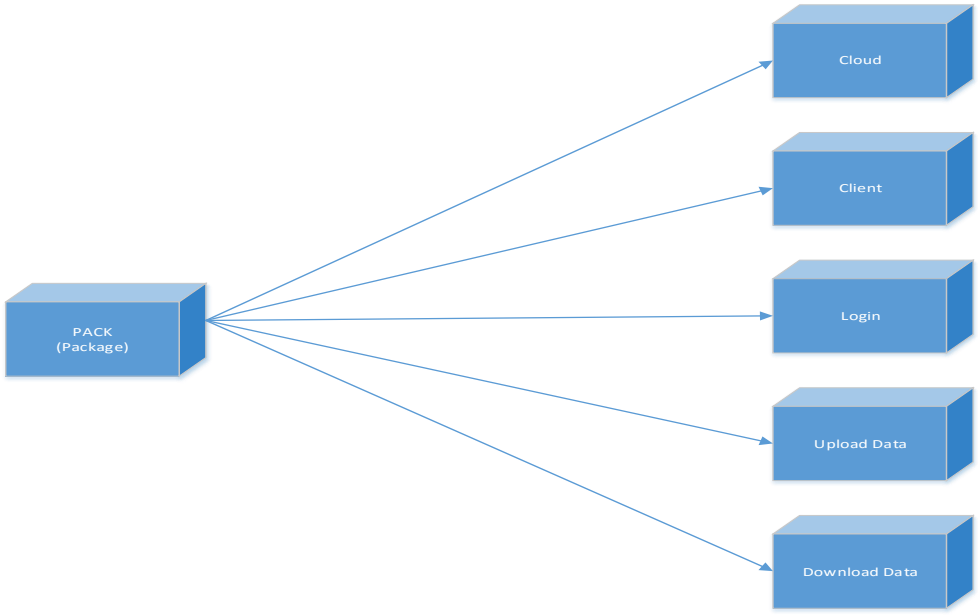

FIGURE 5. Deployment diagram.

The component diagram is the representation of the connection between the various different components in a system. Generally, the component diagram depicts the connections

made using an assembly connector. The component diagram for this protocol is given in Figure 6.

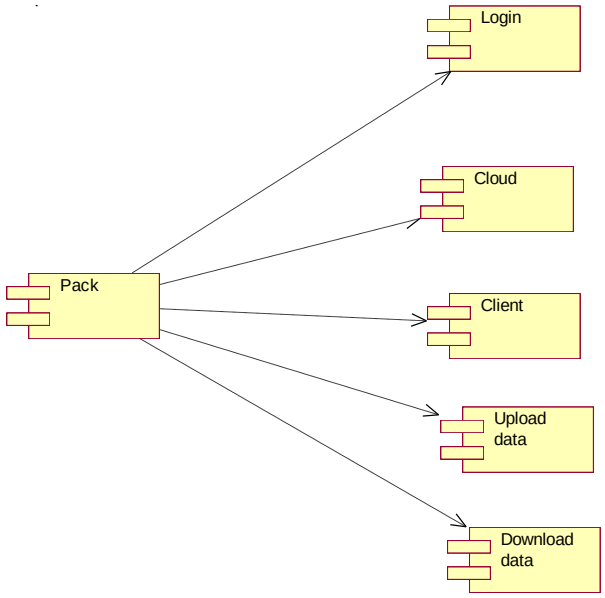

FIGURE 6. Component diagram.

PACK Algorithm

This section describes the receiver based PACK protocol. Optimizations that can be performed in this protocol are discussed in the next section. The basic protocol working is first explained. This is then followed by an in-depth explanation at the receiver and sender.

The data stream received is parsed into a variable sized sequence of content based digitally signed data chunks. These generated chunks are compared with the chunks present in the local storage unit called the chunk store. If a chunk match is successful then the subsequent sequence of data chunks, called chain, are retrieved with the help of the least recently used (LRU) chunk pointers which is included in the chunk metadata. A prediction is then sent to the sender for subsequent data. This sent prediction is based on the constructed chain. Hint, which is part of the prediction, is an easy to calculate function. Hint has a small false positive value.

The prediction sent includes predicted data range, the hint and the digital signature of the data chunk. Upon reception of the prediction the sender first verifies the range in its buffered data. The hint is then verified. If these two match, then SHA-1 algorithm is applied to match the signature of the data chunk received. SHA-1 is a hashing algorithm used for generating digital signatures. This algorithm generates a 160 bit long digital signature. If the signature matches, then an acknowledgement message is sent to the receiver. This confirmation message asks the receiver to copy the data from the chunk store.

Receiver Chunk Store: The chunk store at the receiver contains the chunk's metadata. Each data chunk's metadata consists of the chunk's digital signature and the pointer pointing to the subsequent chunk in the last received data stream. When a new data is received, it gets parsed into chunk's. The digital signature of each chunk is calculated using SHA-1. The data chunk and the signature are added to the chunk store. Along with these the metadata of the chunk received previously in the same data stream is updated in the chunk pointer. All the data chunks produced have a small size. This ensures better redundancy elimination when there are data modifications. Smaller data size also ensures a better storage index size, memory and magnetic disk usage.

Receiver Algorithm: The digital signature is calculated whenever a new data chunk is received. This computed signature is then compared to the signatures that are present in the chunk store. If a match is found, then the receiver scans the chunk's metadata to determine if it is part of a previously received data chain. If it is part of the previously received data chain a prediction is sent to the sender for the next few expected data chunks. The prediction comprises of the offset of the byte stream along with the identity of the subsequent chunks (PRED command). Upon receiving a prediction, the sender responds with a PRED-ACK message.

Upon successfully receiving this confirmation message the data is copied from the receiver cache to the TCP buffer. The data is placed in accordance with its sequence number. The receiver then sends a normal TCP acknowledgement along with the next TCP sequence number. If the prediction is false, then the normal receiver operation is continued without the exchange of the PRED-ACK message.

Sender Algorithm: After the reception of the PRED message from the receiver the sender matches these predictions to the buffered (yet to be sent) data. For each received prediction the hint is verified and the TCP sequence is determined. Once a hint match is found the digital signature of the corresponding data chunk is calculated using SHA-1. If the signature matches the received signature, then it can be safely confirmed that the prediction is correct. The outgoing data is then replaced by the PRED-ACK message. Figures 7 and 8 give the state machine diagram and the flow chart of the sender.

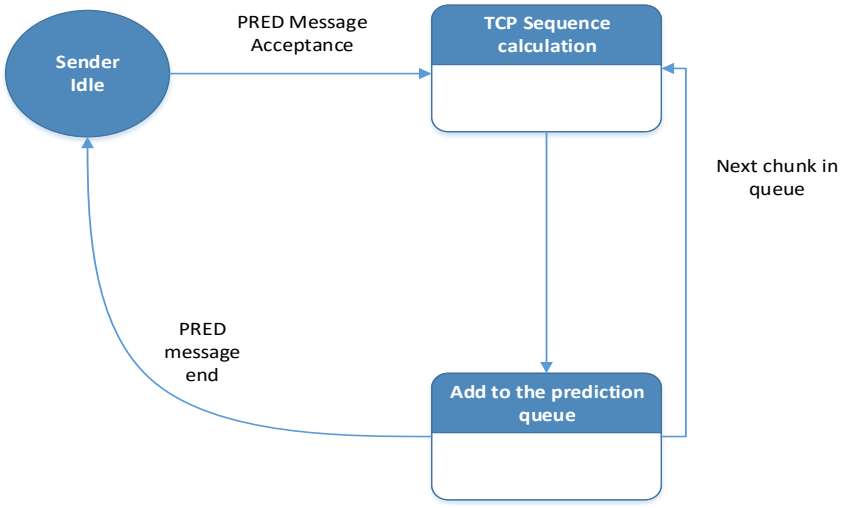

FIGURE 7. Prediction queue filling.

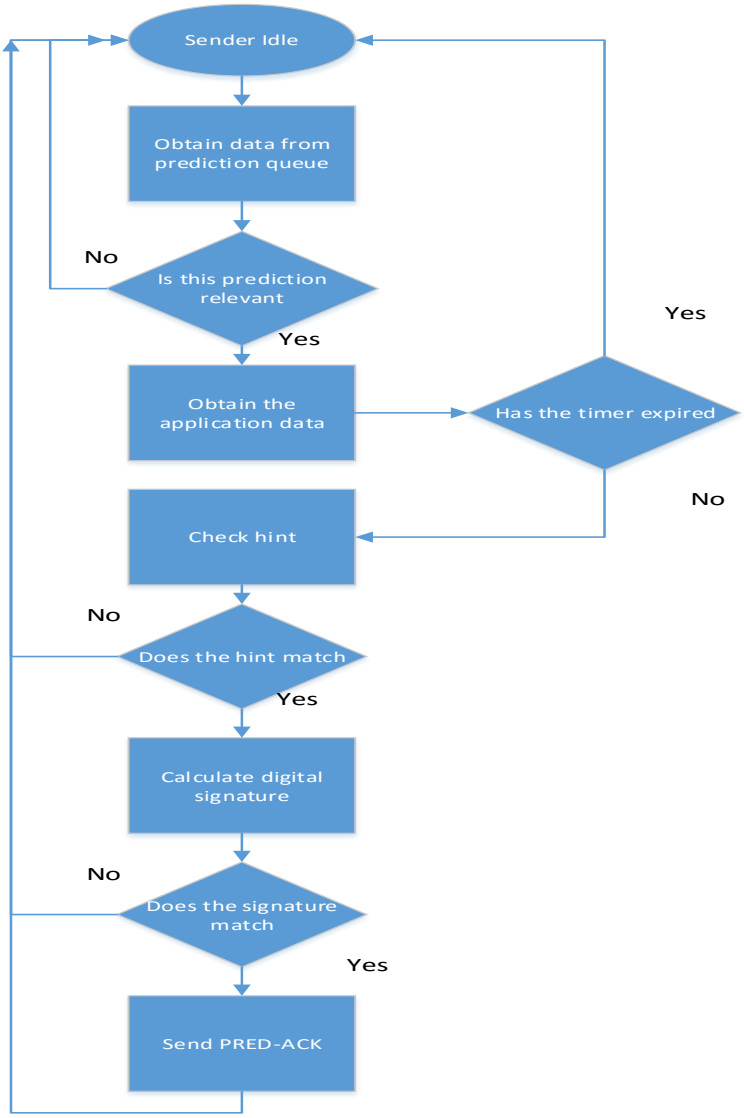

FIGURE 8. Process of analysis of the prediction queues.

Wire protocol: PACK protocol is carried by the TCP options field. This operation minimizes the overheads and also helps in conforming to the existing firewalls. The PACK protocol operates on the assumption that the data to be transmitted is redundant. During the initial handshake the PACK option field is activated by both the sender and the receiver by adding a PACK permitted flag to the TCP options field. One or more TCP segments are then used for sending the redundant data. The receiver then checks if the chunk received matches a chunk in the chunk store. The receiver then sends the prediction and TCP ACK in the options

field. A PRED-ACK message is then sent in place of data by the sender. The PACK protocol for a wire is given in Figure 9.

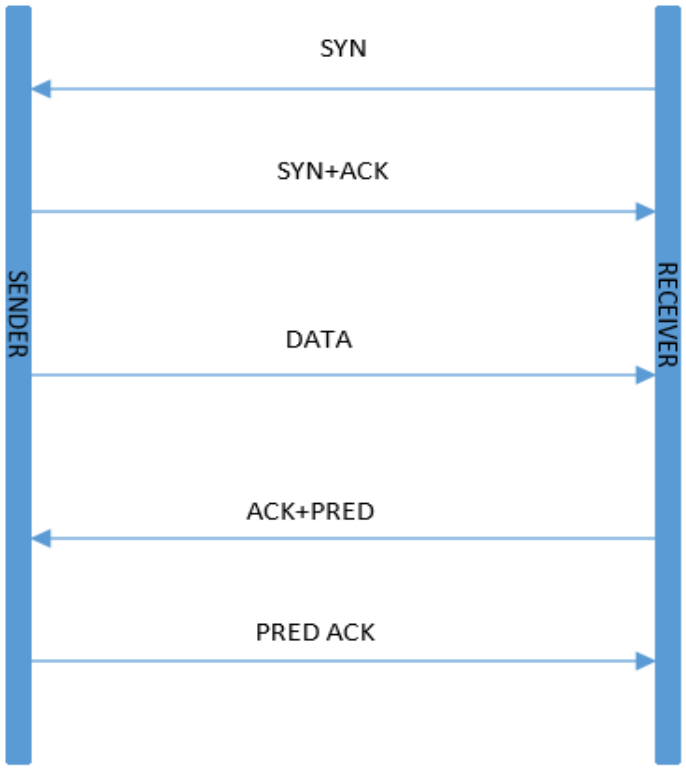

FIGURE 9. Messages exchanged in a wire.

Implementation

Java is the programming language used for this project implementation. Six modules have been created for implementing this project. The six modules are: register module, login module, upload module, client module, query details module and pack module. The register module is used by the client to register on the cloud network by providing username, password and other details. This module is used for a new user to create an account on the cloud network. This module is also created as a Graphical User Interface (GUI). After registration the login

module is used to log the client into the server. Once the user has successfully logged himself/herself into the server he/she has to upload the data with the help of upload module.

With the help of this module the file to be uploaded can be chosen. The fourth module is the client module. This module comprises of the search category and search query. Query module is used to display the queries. This module interacts with the SQL server to obtain the user name and password information. This is used to verify if the user is rightfully registered on the cloud. PACK module is the sixth module that was designed for this project and is the main module of the project. This takes care of the PACK protocol. This module consists of registration, login and upload file details. These modules comprise of a series of classes.

Once the code was written it was tested. There are different types of testing that can be done. The second testing is the module testing. In this testing individual modules are tested one after the other. This is done to locate errors in each module. Testing is done in the ascending order. The lowest and smallest module is tested first and the highest and biggest is tested in the end. The module testing is then followed by integration testing. This module testing helps in detecting errors resulting due to integration of different modules. After passing through all these testing methods the system is tested as a complete entity. This is termed as acceptance test.

CHAPTER 4

OPTIMIZATION AND FUTURE ADVANCEMENTS

This project deals with the basic protocol implementation of PACK. But there are many optimizations that are possible in this protocol. These optimizations are described in this section.

Self-Modifying Receiver Window

Algorithm first checks to see if there is a local copy of data present in the local receiver cache. If there is a local copy of data present, then it is directly downloaded. Downloading of the local copy of data is termed as virtual data. The virtual data rate thus depends on the rate of transmission of prediction messages by the receiver. This can be achieved by either sending several predictions in a message or by expanding the PRED command range. By using this PACK algorithm, a large number of predictions can be combined together and a single hint and a signature can be generated for this entire message. The prediction window can also be varied gradually. It is initially set to a lower value and is increased in steps based on the incoming data. This scheme can be incorporated to take into account the additional data transferred by the sender before the reception of predictions from the receiver. Once this additional data is received by the receiver its prediction can be partially confirmed. Upon getting this confirmation the receiver increases its virtual window size. If there is a disparity detected, then the receiver switches back to the previous window size. In this modification the receiver, after reading the data, replaces the sequence number by one plus the last byte of the redundant data that was just read. It then doubles the size of the virtual window. These techniques will yield a higher data rate.

A Cloud Server Based Receiver

In the future the PACK algorithm can work with the receiver being a cloud server. This is possible when the sender doesn't have any power limitations. The receiver cache has to be maintained in the cloud server.

Receiver or Sender Hybrid Mode

The PACK protocol explained in this report deals with the algorithm implemented at the receiver. This approach is suitable if the data is continuous. The efficiency of this approach reduces remarkably if the data is scattered. In case of scattered data, the predictions are constantly interrupted. This forces the receiver to use the network for data downloading till a data match is found and reported to the sender. This sort of data can be handled more efficiently if the algorithm works on the sender. Thus in the future the efficiency can be increased by algorithm working in both a sender/receiver mode. Sender mode should be established with a negligible buffering overhead. Also, the computational cost involved should be minimal. By this advancement the receiver first assesses and reports the situation back to the sender. The sender then reassesses it and decides the operating mode. For this advancement to be successful an additional command which tracks the dispersion should be introduced.

CHAPTER 5

RESULTS AND DISCUSSIONS

In this report the cloud server was created on the D drive. Every user needs a unique login id and password to upload files to the cloud. The front end view of the designed client looks as the Figure 10 given below.

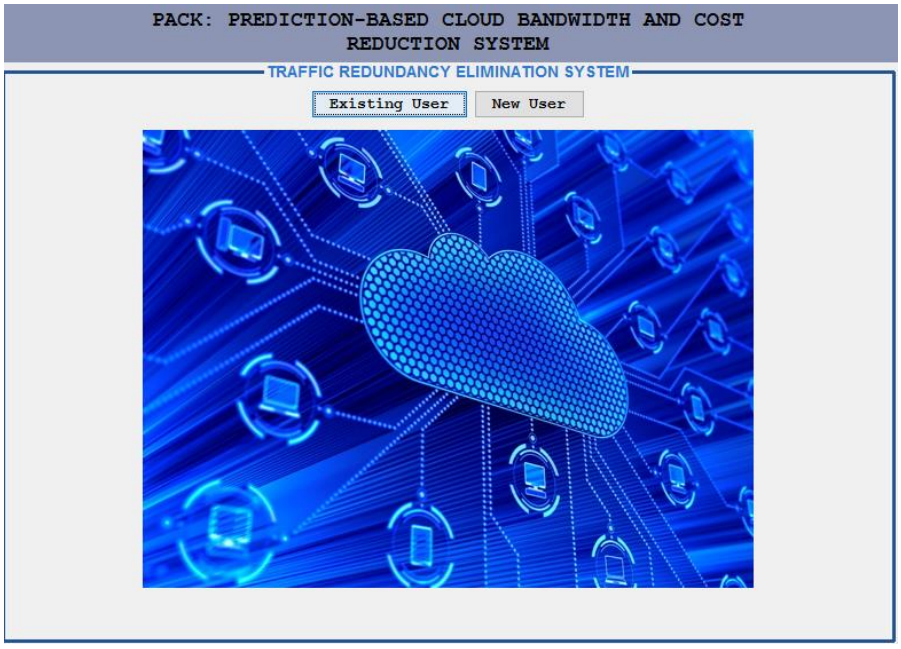

FIGURE 10. Client window.

Every user is directed to the login page where the user can either create a new login or can login using a previously created account. A folder is created corresponding to every registered user on the cloud server. Every file uploaded by the user gets saved in the user's folder.

A user can download only the files that he/she has uploaded from the cloud. The file can be selected from the drop down menu containing all the files selected. The file is downloaded in chunks. Every file is divided into variably sized data chunks. The number of data chunks the file is getting divided into is also shown in the client server window. The receiver checks in the

receiver cache if this data chunk was previously received. If the data chunk is already present in the receiver chunk a prediction message is sent to the sender. It is indicated by the phrase “copy” or “no copy” on the client side. Once the sender confirms that this information the data chunk and the subsequent data chunks are downloaded from the chunk store. For instance, when a registered user is downloading a file containing the text “hello good morning” the file is divided into two parts. Since the file was previously downloaded a local copy of the file is retrieved from the receiver cache. This action is indicated by the term “copy” in the client window. This is given in Figure 11.

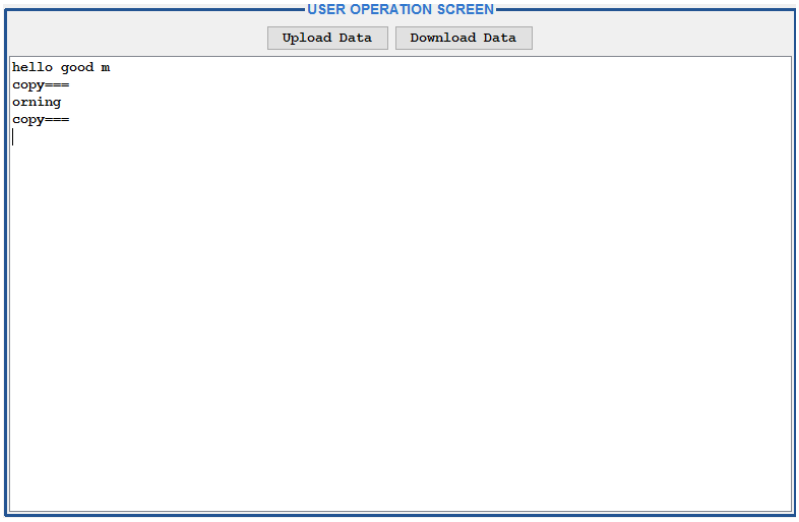

FIGURE 11. File split on client.

The cloud side of the window matching is shown in Figure 12. Here a local copy of the data is retrieved from the local cache. The differentiation between the newly received and the redundant data can be seen by the graph. The graph generated looks like Figure 13.

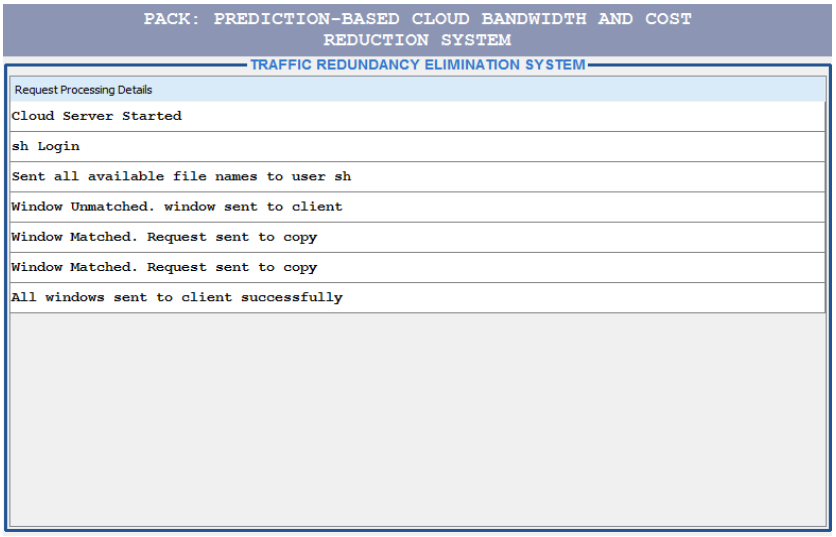

FIGURE 12. Window matching action on cloud.

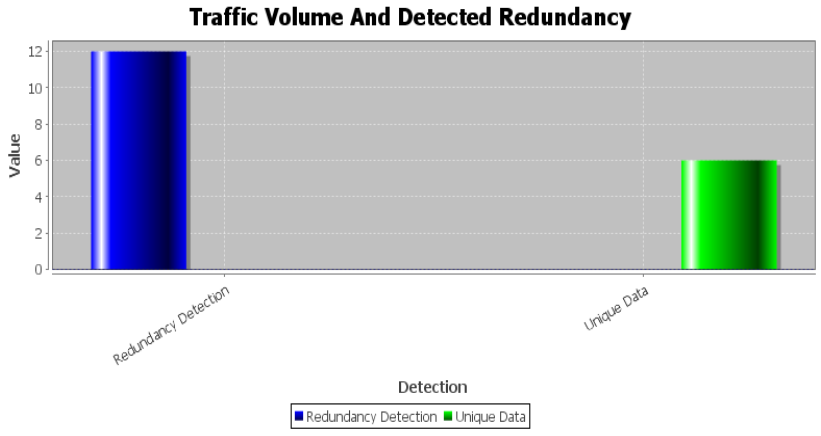

FIGURE 13. Traffic volume and detected redundancy.

CHAPTER 6

CONCLUSION

Cloud computing technology is becoming popular nowadays. This popularity is expected to increase drastically over time. In order to meet this growing demand, an effective TRE algorithm is essential. The existing middle box solutions are not suitable for a cloud network as they mainly focus on reducing the network payment for a single user. In a cloud environment the user pays only for the resources used. Thus, making middleboxes inefficient.

This report deals with the design and implementation of a cloud compatible protocol termed PACK. The overall operational cloud cost is reduced significantly in this design. This design also minimizes the application latency between the sender and the receiver. PACK protocol doesn't require the server to constantly retain the client's status. This feature helps in enabling the cloud elasticity and user mobility. This novel TRE method is capable of eliminating the redundancy, without the three-way handshake application, on the data arriving from multiple servers. Due to the higher data rate achieved, this protocol performs much better than the sender based TRE. This protocol also requires an additional effort on the sender only if the redundancy feature is exploited. This aids in further reducing the operational cost of the cloud.

Three future extensions are possible in this project. The first one is the use of a single prediction message to send multiple data chunk predictions. The digital signature of this prediction is calculated using the SHA-1 method. This feature will increase the rate at which the data is transmitted between the client and the cloud. This rate is termed as the virtual data rate. The second extension possible is to make the cloud server a receiver. This extension will help in conservation of the upstream bandwidth. The working of the algorithm remains unchanged. The cloud server should have a cache for this feature to work. The third extension is the operation in

a hybrid mode. This mode allows the operation of the protocol in either the sender or receiver driven mode. This mode is useful when the data is scattered.

Thus, the new TRE developed helps in managing the traffic between the client and the cloud. As this method is receiver-based it has a higher efficiency compared to the existing TRE techniques.

REFERENCES

REFERENCES

[1] E. Zohar, I. Cidon, and O. Mokryn, "PACK: Prediction-based cloud bandwidth and cost reduction system," *IEEE/ACM Transactions on Networking,* vol. 22, pp.1, Feb 2014.

[2] E. Zohar, I. Cidon, and O. Mokryn, "The power of prediction: Cloud bandwidth and cost reduction," in *Proc. Association for Computer Machineries Special Interest Group on Data Communication (SIGCOMM)*, 2011, pp. 86–97.

[3] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. Association for Computing Machinery (ACM)*, vol. 53, pp. 50–58, 2010.

[4] U. Manber, "Finding similar files in a large file system," in *Proc. The Advanced Computer Systems Association (USENIX) Winter Tech. Conf.*, 1994, pp.1–10.

[5] N. T. Spring and D. Wetherall, "A protocol-independent technique for eliminating redundant network traffic," in *Proc. Association for Computer Machineries Special Interest Group on Data Communication (SIGCOMM)*, 2000, vol. 30, pp. 87–95.

[6] A. Muthitacharoen, B. Chen, and D. Mazières, "A low-bandwidth network file system," in *Proc. Symposium on Operating Systems Principles (SOSP)*, 2001, pp. 174–187.

[7] E. Lev-Ran, I. Cidon, and I. Z. Ben-Shaul, "Method and apparatus for reducing network traffic over low bandwidth links," US Patent 7636767, Nov. 2009.

[8] S. Mccanne and M. Demmer, "Content-based segmentation scheme for data compression in storage and transmission including hierarchical segment representation," US Patent 6828925, Dec. 2004.

[9] R. Williams, "Method for partitioning a block of data into subblocks and for storing and communicating such subblocks," US Patent 5990810, Nov. 1999.

[10] A. Anand, C. Muthukrishan, A. Akella and R. Ramjee, "Redundancy in network traffic: Findings and implications," in *Proc. Association for Computer Machinery's Special Interest Group on Measurement and Evaluation (SIGMETRICS)*, 2009, pp. 37-48.

[12] A. Anand, V. Sekar and A. Akella, "SmartRE: An architecture for coordinated network-wide redundancy elimination," in *Proc. Association for Computer Machineries Special Interest Group on Data Communication (SIGCOMM)*, 2009, vol. 39, pp.87-89

[13] Oracle Java 1.6. [Online]. Available: http://docs.oracle.com/javase/6/docs/api/