Indian Journal of Emerging Electronics in Computer Communications-IJEECC
Vol.1, Issue.1, pp.116-122 (2014), ISSN: 2393-8366

Implementation of Common Caching In a Standalone Cloud Based System

Geno Bennet Singh[#1], Mounish Kishan[#2], T.Shantha Kumar[*3], Dr.D.C.Joy Winnie Wise[*4]

[#1,2]U.G Scholar, [*3]Asst, Professor[*4] ,HOD/Dean,
Department of Computer Science and Engineering,
Alpha College of Engineering, Chennai, T.N, India
[1]*genobennetsingh274@gmail.com*[2]*mounish@2ndeye.com*,

Abstract: This paper introduces Cloud computing can and does mean different things to different people. The common characteristics most share are on demand scalability of highly available and reliable pooled computing resources, secure access to metered services from nearly anywhere, and dislocation of data from inside to outside the organization. While aspects of these characteristics have been realized to a certain extent, cloud computing remains a work in progress. This project deals with the security and privacy challenges pertinent to public cloud computing and reduces the network traffic problem while many persons accessing the same file. And also points out considerations organizations should take when outsourcing data, applications, and infrastructure to a public cloud environment.

Keywords: *Dislocation of data, Virtual cache, SWNET.*

1. INTRODUCTION

RECENT emergence of data enabled mobile devices and wireless-enabled data applications have fostered new content dissemination models in today's mobile Ecosystem. A list of such devices includes Apple's iPhone, Google's Android, Amazon's Kindle, and electronic book readers from other vendors. The array of data applications includes electronic book and magazine readers and mobile phone Apps. The level of proliferation of mobile applications is indicated by the example fact that as of October 2010, Apple's App Store offered over 100,000 apps that are downloadable by the smart phone users. With the conventional download model, a user down- loads contents directly from a Content Provider's (CP) server over a Communication Service Provider's (CSP) network. Downloading content through CSP's network involves a cost which must be paid either by end users or by the content provider. In this work, we adopt Amazon Kindle electronic book delivery business model in which the CP (Amazon), pays to Sprint, the CSP, for the cost of network usage due to downloaded e-books by Kindle users.

When users carrying mobile devices physically gather in settings such as University campus, work place, Mall, Airport and other public places, Social Wireless Networks (SWNETs) can be formed using ad hoc wireless connections between the devices. With the existence of such SWNETs, an alternative approach to content access by a device would be to first search the local SWNET for the requested content before downloading it from the CP's server. The expected content provisioning cost of such an approach can be significantly lower since the download cost to the CSP would be avoided when the content is found within the local SWNET. This mechanism is termed as cooperative caching. In order to encourage the End-Consumers (EC) to cache previously downloaded content and to share it with other end-consumers, a peer-to-peer rebate mechanism is pro- posed. This mechanism can serve as an incentive so that the end-consumers are enticed to participate in cooperative content caching in spite of the storage and energy costs. In order for cooperative caching to provide cost benefits, this peer-to-peer rebate must be dimensioned to be smaller than the content download cost paid to the CSP.

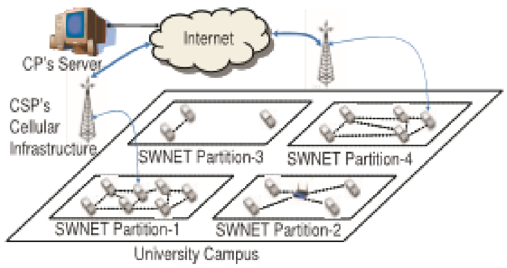


Fig. 1. Content access from an SWNET in a University Campus.

This rebate should be factored in the content provider's overall cost. Due to their limited storage, mobile handheld devices are not expected to store all downloaded content for long. This means after downloading and using a purchased electronic content, a device may remove it from the storage. For example in Amazon Kindle clients (iPhone, iPad, etc.) an archive mode is available using which a user simply removes a book after reading it, although it remains archived as a purchased item in Amazon's cloud server. Under the above pricing and data storage model a key question for cooperative caching is: How to store contents in nodes such that the average content provisioning cost in the network is minimized?

2. CACHING FOR OPTIMAL OBJECT PLACEMENT

Split Cache Replacement to realize the optimal object placement under homoge- neous object request model we propose the following Split Cache policy in which the available cache space in each device is divided into a duplicate segment fraction and a unique segment. In the first segment, nodes can store the most popular objects without worrying about the object duplication and in the second segment only unique objects are allowed to be stored. The parameter in indicates the fraction of cache that is used for storing duplicated objects. With the Split Cache replacement policy, soon after an object is downloaded from the CP's server, it is categorized as a unique object as there is only one copy of this object in the network. Also, when a node downloads an object from another SWNET node, that object is categorized as a duplicated object as there are now at least two copies of that object in the network. For storing a new unique object, the least popular object in the whole cache is selected as a candidate and it is replaced with the new object if it is less popular than the new incoming object. For a duplicated object, however, the evictee candidate is selected only from the first duplicate segment of the cache. In other words, a unique object is never evicted in order to accommodate a duplicated object. The Split Cache object replacement mechanism realizes the optimal strategy established in Section 4. With this mechanism, at steady state all devices' caches maintain the same object set in their duplicate areas, but distinct objects in their unique areas. The pseudo code of Split Cache replacement policy is shown in Algorithm 1.

INPUT: Object O_{new}
IF (O_{new} *is downloaded from another node*)
    O_{min} = *the least popular obj in the* ***duplicate*** *area*
ELSE
    O_{min} = *The least popualar obj in the* ***entire*** *cache*
END
IF (O_{nel} *.popularity* > O_{min}*.popularity*)
    replace O_{min} *l ith* O_{nel}

Algorithm 1: Split Cache object replacement policy

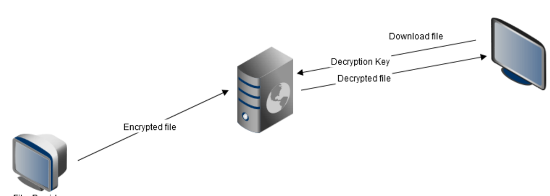
2.1ANDROID SWNET TESTBED

The Split Cache protocol was also implemented as an Android App on a seven-phone Social Wireless Network. Based on Zipf distribution over 5,000 objects, each node was programmed to generate 1 request per second. The requests are homogeneous in these experiments. Each phone is able to store up to 50 different objects in its local cache (i.e., C ¼ 50). After generating a request for an object, a phone first checks its local cache and if its local search fails, it searches the object in the other six phones using an ad hoc Wi-Fi network acting as the interphone peer-to-peer links. If the node does not receive a reply within two seconds after sending the request, it downloads the object directly from a desk-top machine that emulates the CP's server. Note that any object downloaded directly from the CP's server is considered as a unique object and it is stored in the unique area of the cache. Fig. 10a reports object provisioning costs from both the analytical expressions and from the tested when varies between 0 and 1, and the rebate to cost ratio is set to 0.5. The cost is analytically computed according to when the parameters m, C, and are set to 7, 50, and 0.8, respectively. Observe that although the costs obtained from the tested maintain values and trends very similar to those from the equation, they are always slightly higher. These differences stem from undesired object duplication as a result of search inaccuracy as follows: when two or more nodes register cache misses at the same time for a supposedly unique object, all of them may attempt to download the object from the CP's server. This can result in undesired object duplications, causing an effective which is larger than the target opt. Such faulty duplication was also found to happen due to erroneous object search in the events of lost search requests in the Wi-Fi phone network. The impacts of undesired object duplication are higher local hit rates and lower remote hit rates (compared to the equation) and therefore higher provisioning costs. It should be also observed that the higher costs due to undesired object duplication happens more when is small. This is because when is very small, the local hit rate is very low. Thus, the number of search requests to the other nodes is quite high. As a result, the absolute number of simultaneous requests and lost search requests as described above are also high. These cause more frequent erroneous object duplications and subsequently higher cost.

3. EXISTING SYSTEM

With the existence of such SWNETs, an alternative approach to content access by a device would be to first search the local SWNET for the requested content before downloading it from the CP's server. The expected content provisioning cost of such an approach can be significantly lower since the download cost to the CSP would be avoided when the content is found within the local SWNET. This mechanism is termed as cooperative caching. In order to encourage the End-Consumers (EC) to cache previously downloaded content and to share it with other end-consumers, a peer-to-peer rebate mechanism is proposed. This mechanism can serve as an incentive so that the end-consumers are enticed to participate in cooperative content caching in spite of the storage and energy costs. In order for cooperative caching to provide cost benefits, this peer-to-peer rebate must be dimensioned to be smaller than the content download cost paid to the CSP. This rebate should be factored in the content provider's overall cost.

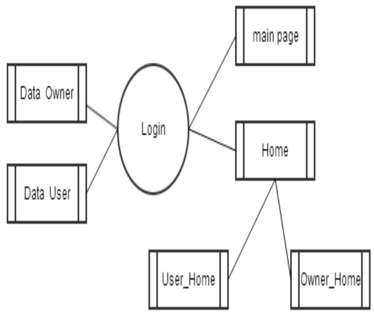
Fig: Existing System



4. MODULES

4.1. User Authentication Module

The user authentication deliberately addresses the user takes place in the project. There are two kinds of users takes place here, one is admin he can upload files and monitors the users and can download the files and also can view the file download history. Another user is normal user any one can register as user and can upload and downloads file.



DFD Level 1

4.2. File Encryption Module

In this module advanced encryption method is used to secure the file. Here we are using Base64 Encryption technique. This encryption process is done to secure the file while upload in cloud. By providing right decryption key only the file downloaded in correct format for others the file will downloaded in corrupted format only.

Fig: File Encryption System

4.3. File Upload Module

In file upload module the user can upload file in the cloud any registered user can upload their files and they will encrypt their file and upload it for the security purpose. Once the file uploaded anyone can download the file but they need to provide the decryption key to download the file.

Fig: File Upload System

4.4. File Download Module

Once the file uploaded any one can download the file but they need to provide the decryption key to download the file with right decryption key only the file downloaded in correct format for others the file will downloaded in corrupted format only. For the first time download only the user will download the file from main Cloud as Mandatory.

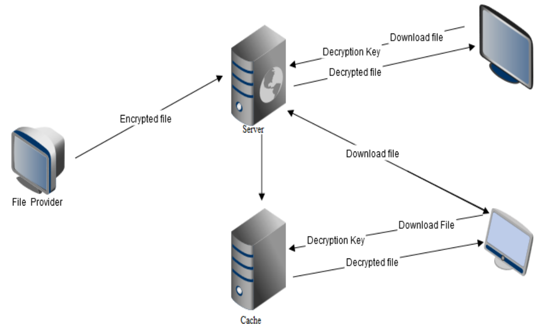
Fig: File Download System

4.5. Virtual Cloud Module

Once the file downloaded in main cloud it will automatically available in virtual cloud here virtual cloud is nothing but the temporary memory or cache memory. After the first time download any user tries to download the file from main memory in the sense the notification will appear to denote that the file is availed in virtual cloud.

4.6. Comparison Module

The user can download the file from both the main cloud and virtual cloud. Comparing with main cloud download the virtual cloud download will takes less time and cost. The user can identify it by comparing the download time taken for same file from main cloud and virtual cloud.



5. CONCLUSION

We Proposed system any one can upload their files as a secured encrypted file and it is necessary to provide the decryption key to download such file. With wrong decryption key the file downloaded as corrupted file only. Once the file downloaded first time it is available in virtual cloud also any one can download the particular file in the virtual cloud it avoids the network traffic and the file available in virtual cloud should take lesser time and cost the main cloud while downloading. In this paper drawing motivation from Amazon's Kindle electronic book delivery business, this paper develops practical network, service, and pricing models which are then used for creating two object caching strategies for minimizing content provisioning costs in networks with homogenous and heterogeneous object demands. The paper constructs analytical and simulation models for analyzing the proposed caching strategies in the presence of selfish users that deviate from network-wide cost-optimal policies. It also reports results from an Android phone based prototype SWNET, validating the presented analytical and simulation results.

REFERENCES

[1] M. Zhao, L. Mason, and W. Wang, "Empirical Study on Human Mobility for Mobile Wireless Networks," Proc. IEEE Military Comm. Conf. (MILCOM), 2008.

[2] "Cambridge Trace File, Human Interaction Study," http:// www.crawdad.org/download/cambridge/haggle/Exp6.tar.gz, 2012.

[3] E. Cohen, B. Krishnamurthy, and J. Rexford, "Evaluating Server- Assisted Cache Replacement in the Web," Proc. Sixth Ann. European Symp. Algorithms, pp. 307-319, 1998.

[4] S. Banerjee and S. Karforma, "A Prototype Design for DRM Based Credit Card Transaction in E-Commerce," Ubiquity, vol. 2008, 2008.

[5] L. Breslau, P. Cao, L. Fan, and S. Shenker, "Web Caching and Zipf- Like Distributions: Evidence and Implications," Proc. IEEE INFOCOM, 1999.

[6] C. Perkins and E. Royer, "Ad-Hoc On-Demand Distance Vector Routing," Proc. IEEE Second Workshop Mobile Systems and Applica- tions, 1999.

[7] S. Podlipnig and L. Boszormenyi, "A Survey of Web Cache Replacement Strategies," ACM Computing Surveys, vol. 35, pp. 374- 398, 2003.

[8] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, "Impact of Human Mobility on Opportunistic Forwarding Algorithms," IEEE Trans. Mobile Computing, vol. 6, no. 6, pp. 606-620, June 2007.

[9] "BU-Web-Client - Six Months of Web Client Traces," http:// www.cs.bu.edu/techreports/1999-011-usertrace-98.gz, 2012.

[10] A. Wolman, M. Voelker, A. Karlin, and H. Levy, "On the Scale and Performance of Cooperative Web Caching," Proc. 17th ACM Symp. Operating Systems Principles, pp. 16-31, 1999.

[11] S. Dykes and K. Robbins, "A Viability Analysis of Cooperative Proxy Caching," Proc. IEEE INFOCOM, 2001.

[12] M. Korupolu and M. Dahlin, "Coordinated Placement and Replacement for Large-Scale Distributed Caches," IEEE Trans. Knowledge and Data Eng., vol. 14, no. 6, pp. 1317-1329, Nov. 2002.

[13] L. Yin and G. Cao, "Supporting Cooperative Caching in Ad Hoc Networks," IEEE Trans. Mobile Computing, vol. 5, no. 1, pp. 77-89, Jan. 2006.

[14] Y. Du, S. Gupta, and G. Varsamopoulos, "Improving On-Demand Data Access Efficiency in MANETs with Cooperative Caching," Ad Hoc Networks, vol. 7, pp. 579-598, May 2009.

[15] C. Chow, H. Leong, and A. Chan, "GroCoca: Group-Based Peer- to-Peer Cooperative Caching in Mobile Environment," IEEE J. Selected Areas in Comm., vol. 25, no. 1, pp. 179-191, Jan. 2007.