Research Article Impact Factor: 0.621 ISSN: 2319-507X
Kiran Deshmukh, IJPRET, 2014; Volume 2 (8): 132-137 IJPRET

INTERNATIONAL JOURNAL OF PURE AND APPLIED RESEARCH IN ENGINEERING AND TECHNOLOGY

A PATH FOR HORIZING YOUR INNOVATIVE WORK

# AN EFFICIENT PACKET ROUTING MECHANISM IN WIRELESS MOBILE NETWORKS USING DCIM

**MR. KIRAN DESHMUKH[1], PROF. R. V. SHAHABADE[2]**

1. ME Student Second Year Computer Engineering, Terna engg. College Nerul, Navi-Mumbai.
2. Assistant Professor, Terna engg. College Nerul, Navi-Mumbai.

**Accepted Date: 15/02/2014 ; Published Date: 01/04/2014**

**Abstract:** This paper proposes distributed cache invalidation mechanism (DCIM), a client-based cache consistency scheme that is implemented on top of a previously proposed architecture for caching data items in mobile ad hoc networks (MANETs).This project deals with an efficient data routing in wireless mobile networks that overcomes the problem of data loss during a very low connectivity range. We introduce DCIM that is totally client-based. DCIM is a pull-based algorithm that implements adaptive time to live (TTL), piggybacking, and prefetching, and provides near strong consistency capabilities. In this paper, DCIM is analyzed to assess the delay and bandwidth gains (or costs) when compared to polling every time and push-based schemes. DCIM was also implemented using ns2, and compared against client-based and server-based schemes to assess its performance experimentally. The consistency ratio, delay, and overhead traffic are reported versus several variables, where DCIM showed to be superior when compared to the other systems.

**Keywords:** TTL, Piggybacking, MANET, DCIM, Perfecting, Invalidation.

**Corresponding Author: Mr. KIRAN DESHMUKH**

**Access Online On:**

www.ijpret.com

**How to Cite This Article:**

Kiran Deshmukh, IJPRET, 2014; Volume 2 (8): 132-137

***PAPER-QR CODE***

## INTRODUCTION

MOBILE devices are the building blocks of mobile ad-hoc networks (MANETs). They are typically characterized by limited resources, high mobility, transient availability, and lack of direct access to the data source (server). In MANET environments, data caching is essential because it increases the ability of mobile devices to access desired data, and improves overall system performance. In a typical caching architecture, several mobile devices cache data that other devices frequently access or query. Data items are essentially an abstraction of application data that can be anything ranging from database records, webpages, ftp files, etc. The major issue that faces client cache management concerns the maintenance of data consistency between the cache client and the data source . All cache consistency algorithms seek to increase the probability of serving from the cache data items that are identical to those on the server. However, achieving strong consistency, where cached items are identical to those on the server, requires costly communications with the server to validate (renew) cached items, considering the resource limited mobile devices and the wireless environments they operate in. Consequently there exist different consistency levels describing the degree to which the cached data is up to date. These levels, other than strong consistency, are weak consistency, delta consistency, probabilistic consistency, and probabilistic delta consistency.[3][6]

In probabilistic delta consistency, a certain cached item is at most *delta* units of time stale with a probability not less than *p* The cache consistency mechanisms in the literature can be grouped into three main categories: push based, pull based, and hybrid approaches .*Push-based* mechanisms are mostly server-based, where the server informs the caches about updates, whereas *Pull-based* approaches are client-based, where the client asks the server to update or validate its cached data.

In this work, we propose a pull-based algorithm that implements adaptive TTL, piggybacking and prefetching , and provides near strong consistency guarantees. Cached data items are assigned adaptive TTL values that correspond to their update rates at the data source.[3] This is the first complete client side approach employing adaptive TTL and achieving superior availability, delay, and traffic performance.[3][6]

### 1.1 Objectives

The main aim of the project is to reduce the delay and data loss due to low connectivity range and hence reduce the burden of the server to know about their caches.

### 1.2 Description

DCIM is a client-based cache consistency scheme that is implemented on top of a previously proposed architecture for caching data items in MANETs, namely COACS, where special nodes cache the queries and the addresses of the nodes that store the responses to these queries. We introduce DCIM that is totally client-based. DCIM is a pull-based algorithm that implements adaptive TTL and provides near strong consistency capabilities. Cached data items are assigned adaptive TTL values that correspond to their update rates at the data source, where items with expired TTL values are grouped in validation requests to the data source to refresh them, whereas un expired ones but with high request rates are pre-fetched from the server.

### 1. Proposed system

In this paper, we propose a pull-based algorithm that implements adaptive TTL, piggybacking, and prefetching, and provides near strong consistency guarantees. Cached data items are assigned adaptive TTL values that correspond to their update rates at the data source.[3][6]. Expired items as well as non-expired ones but meet certain criteria are grouped in validation requests to the data source, which in turn sends the cache devices the actual items that have changed, or invalidates them, based on their request rates. This approach, which we call distributed cache invalidation mechanism (DCIM), works on top of the COACS cooperative caching architecture.

TTL algorithms are popular due to their simplicity, sufficiently good performance, and flexibility to assign TTL values to individual data items.[3][6]Also, they are attractive in mobile environments because of limited device energy and network bandwidth and frequent device disconnections. TTL algorithms are also completely client based and require minimal server functionality. From this perspective, TTL-based algorithms are more practical to deploy and are more scalable. This is the first complete client side approach employing adaptive TTL and achieving superior availability, delay, and traffic performance.[3][6]

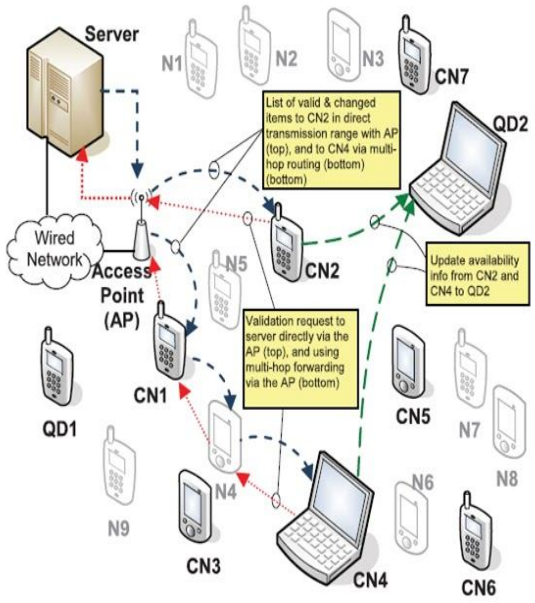


**Fig.1.Overview of DCIM basic design**

**Description:**

The server acts as the vital source of information to which requests are made to receive and update data. In the above figure it is evident that one of mobile nodes has a weaker connectivity range[11][9].Hence in order to overcome the delay and data loss, it forwards the requests through the intermediate node query directory and cache node. Cache node maintains the frequently used data for a temporary memory instance while the query directory maintains the information about those caches maintained in the cache nodes and efficiently directs them to the requesting node. Access point determines the connectivity range of the mobile nodes.[1][2].

**Request Node and Query Directories:**

A node that desires a data item sends its request to its nearest QD. If this QDfinds the query in its cache, it forwards the request to the CN caching the item, which, in turn, sends the item to the requesting node (RN). Otherwise, it forwards it to its nearest QD, which has not received the request yet. If the request traverses all QD's without being found, a miss occurs and it gets forwarded to the server which sends the data item to the RN.

**Requesting Node Sending the Request**

The Requesting Node sends the Request for the particular Data Items. The Request flow through Query Directory to Cache Node.[1][2]

**Server and Cache Node Maintaining:**

The server is the final depot of request. When the data is not found in any ofthe cache nodes the request is sent to the server which responds the request to the requesting cache node that failed to respond to the request nodes request. Whenever the data in the server is changed, it must be reflected in on the corresponding cache nodes to avoid the wrong delivery of data. Such data changes are immediately reflected based on the calculated hit rate which says the demand for that particular data. [1][2]

## 2. Result &Discussion

Expected result will be after complete the project work. The cache node isdata's initially, it polls the random data from the server and then maintained for a calculated TTL based on the demand for that data i.e. Number of requests for that particular data that is maintained by that cache node. Whenever a data is requested by the requestnode, hit rate is increased. He created and since it does not contain any guest[7]. The request containing the information about the data is forwarded by the query directory to the list of all the cache nodes in its vicinity. The data is received through the query directory; the response is indicated through the textbox in the response. The cache node maintains the hit rates based on the demand request for that particular data which in this case is the java file.

## 3. Conclusion& Future Scope

The proposed packet routing mechanism applicable in a MANET environment wherein the wireless mobile communication can be handled with less in consistency. The project has reduced the delivery failures in the scenario of low connectivity ranges.

With the use of Distributed Caches the fetching of data has been lot easier and less time consuming. The invalidation of caches that are distributed periodically from the server has been made easier compared to other MANET concepts like SSUM (Short serve update mechanism and COACS (Cooperative Cache-Based Data Access in AdHoc Networks). The cache node is made to poll data from the server periodically and randomly and hence the burden of the server to maintain the state information about the cache nodes is strictly reduced compared to other mechanisms.

In future research this will show, how securely the data transfer can be made within the nodes involved in the scenario. The process that is taking place between the nodes involved can be made to act as background process without the knowledge of the user of the mobile node.

## 4. REFERENCES

1. J. Cao; Y. Zhang, G. Cao, X. Li, "Data Consistency for Cooperative Caching in Mobile Environments," Computer , v.40, n.4, pp.60-66, 2007

2. P. Cao, C. Liu, "Maintaining strong cache consistency in the World-Wide Web," *IEEE Trans. Computers*, v. 47, pp. 445–457, 1998.

3. J. Jung, A.W. Berger, H. Balakrishnan, "Modeling TTL-based internet caches," *IEEE INFOCOM 2003*, San Francisco, CA, March 2003

4. H. Artail, H. Safa, K. Mershad, Z. Abou-Atme, N. Sulieman, "COACS: A Cooperative and adaptive caching system for MANETS", *IEEE TMC*, v.7, n.8, pp. 961-977, 2008

5. K. Mershad, H. Artail, "SSUM: Smart Server Update Mechanism for Maintaining Cache Consistency in Mobile Environments," IEEE TMC, v. 9, n. 6, pp.778-795, 2010.

6. Y. Fang, Z. Haas, B. Liang, Y. B. Lin, "TTL prediction schemes and the effects of inter-update time distribution on wireless data access," Wireless Networks, v. 10, pp. 607-619, 2004.

7. J. Jing, A. Elmagarmid, A. Helal, R. Alonso, "Bit-Sequences: An Adaptive Cache Invalidation Method in Mobile Client/Server Environments," Mobile Networks and Applications, pp. 115-127, 1997.

8. H. Maalouf, M. Gurcan, "Minimization of the update response time in a distributed database system," Performance Evaluation, v. 50, n. 4, pp. 245-66,2002.

9. Z. Wang, S. Das, H. Che, M. Kumar, "A scalable asynchronous cache consistency Scheme (SACCS) for mobile environments," *IEEE TPDS*, v. 15, n.11, pp. 983- 995, 2004.66

10. "Cache Invalidation Scheme for Mobile Computing Systems with Real-time Data" Joe Chun-Hung Yuen, Edward Chan, Kam-Yiu Lam and H. W. Leung

11. [HL97] Hu, Q. and Lee, D.L., "Adaptive Cache Invalidation Methods in Mobile Environments", Proc. 6th IEEE Intl. Symp. On High Performance Distributed Computing Environments, 1997.