Survey: Client Data Cache Invalidation Mechanism in Trust based Wireless Mobile Networks

Amulya T
M.Tech Student, Dept of CS&E
AIT Chickmagalur

Adarsh M J
Asst. Professor, Dept of CS&E
AIT Chickmagalur

ABSTRACT
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 improve overall system performance. In this paper client data cache invalidation mechanism is proposed, it is a client-based cache consistency scheme that is implemented on the top of a previously proposed architecture for caching data items in MANETs, namely cooperative and adaptive caching system (COACS) [16]. It is a special node that cache the queries and the address of the nodes that store the responses to these queries. Previously a server-based consistency scheme i.e., smart server update mechanism (SSUM) was proposed which is server based, where as in this paper client data cache invalidation mechanism is proposed that is totally client-based. Client data cache invalidation mechanism is a pull-based algorithm that implements adoptive time to live (TTL), piggy backing, and pre-fetching, and provides near strong consistency capabilities, client data cache invalidation mechanism is analyzed to assess the delay and bandwidth gains (or costs) when compared to polling every time and push-based scheme.

Index Terms *-Cache consistency, data caching, MANET, TTL*

1. INTRODUCTION
Wireless cellular systems are in use since 1980s. We have seen their evolutions to first, second and third generation's wireless systems [2]. Wireless systems operate with the aid of a centralized supporting structure such as an access point. These access points assist the wireless users to keep connected with the wireless system, when they roam from one place to the other.

The presence of a fixed supporting structure limits the adoptability of wireless systems i.e., the technology cannot work effectively in places where there is no fixed infrastructure. So in order to advance, a new type of wireless systems known as mobile ad hoc networks (MANETs) was proposed. MANET network operate in the absence of fixed infrastructure. It is an autonomous system of mobile nodes connected by wireless links each node operates as an end system and a router for all other nodes in the network. An ad hoc network is a collection of wireless mobile nodes which dynamically forms a temporary network without the aid of any established infrastructure or centralized administration.

Mobile Ad Hoc Networks has become one of the most prevalent areas of research in the recent years [3]. MANET is the new emerging technology which enables users to communicate without any physical infrastructure regardless of their geographical location, that's why it is sometimes referred to as an "infrastructure less" network. The proliferation of cheaper, small and more powerful device make MANET a fastest growing network. Mobile devices are the building blocks of MANET. They are typically characterized by limited resources, high mobility, transient availability, and lack of direct access to the server also called as the data source.

Data caching is one of the essential part of the MANET environment because it increases the ability of mobile devices to access desired data and it also improves the system overall performances [1], [6], [7]. Usually in caching techniques i.e., in a caching architecture, several mobile devices cache data that other devices frequently access or request. Data items that mobile devices cache can be anything ranging from data base records, web pages, ftp files, etc. Maintenance of data consistency between the cache client and the data source is the major issue that the client cache management concerns upon. Hence, all cache consistency algorithms try to increase the probability of serving from the cache data items that are identical to those on the server [5]. To achieve strong consistency, where cached items are identical to those on the server, requires robust communication with the server to renew (validate) cached items and by considering the limited resource mobile devices, wireless environments they operate in. Also, other than the strong consistency there exist different consistency devices that describe the degree to which the cached data is up to date and they are weak consistency, delta consistency, probabilistic consistency and probabilistic delta consistency.

In weak consistency, client queries might get served with inconsistent (stale) data items, while in delta consistency cached (stored) data items are stale for up to a period of times and that period is denoted as delta. In probabilistic consistency, a data item is consistent with the source with a certain probability denoted as p. Finally, 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 can be grouped in to three main categories i.e., push-based, pull-based and hybrid approaches. Push-based mechanisms are mostly server-based, pull-based approaches are client based. In push-based mechanism, server informs the cache about the updates, where as in pull-based Mechanism client asks the server to update or validate its cached data. Finally in hybrid scheme, the server pushes the updates or the client pulls them.

The pull model and the push model designate two well-known approaches for exchanging data between two distant entities. The Magazine metaphor is a simple illustration of these models: if we want to read our favorite magazine every day, you can either go and buy it every week, or subscribe to it once and then receive it automatically at home. The former is an example of pull, the latter of push. The pull model is based on the request/response paradigm (called *data polling*, or simply *polling*, in traditional SNMP-based network management) the client sends a request to the server, then the server answers, either synchronously or asynchronously. This is functionally equivalent to the client "pulling" the data off the server. The push model, conversely, is based on the publish/subscribe/distribute paradigm. In this model, agents first advertise what MIBs they support, and what SNMP notifications they can send the administrator then subscribes the manager (the NMS) to the data he/she is interested in, specifies how often the manager should receive this data, and disconnects. Later on, each agent individually takes the initiative to "push" data to the manager, either on a regular basis via a scheduler (e.g., for network monitoring) or asynchronously (e.g., to send SNMP notifications).

The Time to live (TTL)-based algorithms are the best example of pull based approaches. In TTL algorithm, TTL value is stored alongside each data item 'd' in the cache and 'd' is considered valid until 'T' time units go by, since, the last update. Such algorithms are popular in mobile environments. Due to their simplicity, sufficiently good performance, and flexibility to assign TTL values to individual data items and also because of limited device energy and network bandwidth and frequent device disconnections. TTL algorithms are also completely client-based which require minimal server functionality. From this perspective, TTL-based algorithms are easier to deploy and are more scalable.

In this paper, a pull-based algorithm is introduced that implements adaptive TTL, piggybacking, and pre-fetching, which provides near strong consistency guarantees. Cached data items are assigned adaptive TTL values that correspond to their update rates at the data source. 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 client cache mechanism works on top of the COACS cooperative caching architecture.

In the rest of the paper, Section 2 discusses related work, Section 3 discusses methodology and finally Section 4 finishes the paper with conclusion and future works.

2. RELATED WORK

Much work has been done in relation to cache consistency in MANETs and also pull based approach. The data cache consistency method proposed in [4], describes how caching of frequently accessed data items will be an important technique that will reduce contention on the narrow-bandwidth, wireless channel. The cache individualization strategies will be affected by the disconnection and mobility of the clients. The server may no longer know which clients are currently residing under its cell, and which of them are currently on. So a different cache invalidation strategy was proposed that helped to study the impact of client's disconnection times on their performance. Also, a study was made with different ways to improve the efficiency of the invalidation techniques. The original IR approach is also proposed in this paper which is mainly used in push-based mechanism along with IR, several algorithms have also been proposed. These algorithms include stateless schemes where, the server stores no information about the client caches [7], [8], [9] and state-full approach where the server maintains state information. This paper mainly concentrates on server side modification and over head processing. More crucially, the methods used in this paper require the server to maintain some state information about the MANET, which is costly in terms of bandwidth consumption, especially in highly dynamic environments. Mainly this paper concentrates on push based scheme but client data cache invalidation mechanism on the other hand, belongs to a different class of approaches.

In the subsequent work [10], the paper presents one of the client polling systems where modern distributed systems involving large numbers of non-stationary clients (mobile hosts, MH) connected via unreliable low bandwidth communication channels that are very prone to frequent disconnections. This disconnection may occur because of different reasons, The clients may voluntarily switch off (to save battery power), or a client may be involuntarily disconnected due to its own movement in a mobile network. A mobile computing environment is characterized by slow wireless links and relatively under privileged hosts with limited battery power. Still, when data at the server changes, the client hosts must be made aware of this fast in order for them to invalidate their cache, otherwise the host would continue to answer queries with the cached values returning incorrect data. The nature of the physical medium coupled with the fact that disconnections from the network are very frequent in mobile computing environments demand a cache invalidation strategy with minimum possible over heads. This paper also provides a new client polling cache maintenance scheme called AS. The objective of the proposed scheme is to minimize the over head for the MHS to validate their cache upon reconnection, to allow stateless servers, and to minimize the bandwidth requirement. A cache validation request is initiated according to a schedule determined by the cache.

These are variants of such systems [11][12] that try to achieve strong consistency by validating each data item before being served to a query. In [11], each cache entry is validated when queried using a modified search algorithm, where as in [12] the system is configured with a probability that controls the validation of the data item from the server as the neighbors when required. Although client poll algorithms have relatively low bandwidth consumption, their access delay is high considering that each item needs to be validated upon each request. Client data cache invalidation mechanism, on the other hand, attempts to provide valid items by adapting expiry intervals to update rates and uses pre-fetching to reduce query delay.

Several TTL algorithms which were proposed for MANETs were motivated by web caches research. These include the adaptive TTL method [13]. As the web continues to explode in size, caching becomes increasingly important, with caching, comes the problem of cache consistency. Conventional wisdom holds strong cache consistency, which is too expensive for the web, and weak consistency methods such as TTL are more appropriate. This paper compares three consistency approaches adaptive TTL, polling-every-time and invalidation. The result of this paper shows that the invalidation performs similar to adaptive TTL, in terms of network traffic, average client response times and server CPU loads. Polling-every-time, on the other hand, leads to significantly more network messages and higher response times than adaptive TTL. Thus it is feasible to maintain strong cache consistency for the web and invalidation is the right approach for it. So a two tier lease-augmented invalidation scheme that addresses the scalability issues in invalidation has also been described. There are many limitations in this study. In summary, the above approaches only provide shallow integration of TTL processing into the cache functionality, and none of them gives a complete TTL-based cache consistency scheme for MANETs. Additionally, they do not include mechanisms for reducing bandwidth consumption, which is crucial in MANET environments.

3. METHODOLOGY

Mobile devices are the building blocks of mobile ad hoc networks (MANETs). The system consists of a MANET of wireless mobile nodes interested in data generated at an external data source connected to the MANET using a wired network (e.g., internet) via Wi-Fi Access Points (APs). Nodes that have direct wireless connectivity to an AP act as gateways, enabling other nodes to communicate with the data source using multi-hop communication. For example, if cache node (CN) access the server through a node (N) and also through other cache node (CN), then it is said to be a gateway, which is connected to the internet via the AP. The data exchanged is abstracted by data items. The fig. 1 shows the overview of client data cache invalidation mechanism basic design.

Client data cache invalidation mechanism is a client-side system that is able to scale too many types of provided services. Client data cache invalidation mechanism fits more naturally into the current state of the Internet with the prevailing client/server paradigm, where clients are responsible for pulling the data from the server, which in turn maintains little state information and seldom pushes data to them. Client data cache invalidation mechanism performs similar or better than push-based approaches, while keeping all the processing at the client side with little overhead.

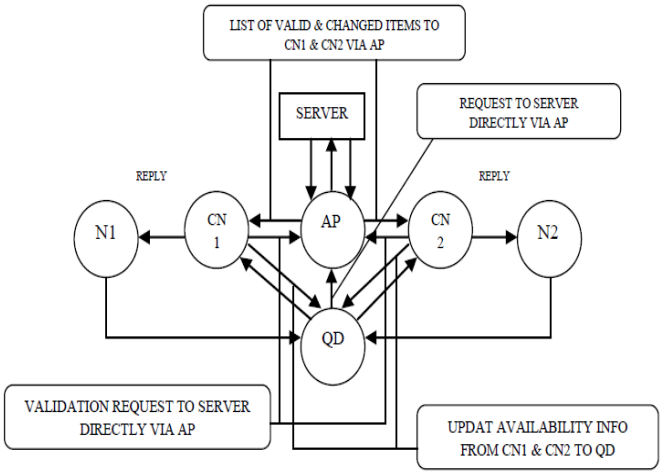

Fig.1. Overview of client data cache invalidation mechanism basic design.

The proposed system builds on top of COACS [15]. The system has three types of nodes: caching nodes (CNs) that cache previously requested items, query directories (QDs) that index the cached items by holding the queries along with the addresses of the corresponding CNs, and requesting nodes (RNs) that are ordinary nodes. Any node, including a QD or a CN, can be a requesting node, and hence, an RN is not actually a special node, as it is only used in the context of describing the system. One, therefore, might view the employed caching system as a two layered distributed database. The first layer contains the QDs which map the queries to the caching nodes which hold the actual items that are responses to these queries, while the second layer is formed by the CNs.

Finally, it is worth mentioning that although our recently introduced SSUM [14] client data cache consistency scheme is also build on COCAS, it is a server-based approach, whereas client data cache invalidation mechanism is completely client-based. The goal of client data cache consistency scheme is to improve the efficiency of the cache updating process in a network of mobile devices which cache data retrieved from a data server, without requiring the latter to maintain state information about the caches. The proposed system is pull-based, where the CNs monitors the TTL information and accordingly triggers the cache updating and validation process.

Client data cache invalidation mechanism is scalable by virtue of the CNs whose number can increase as the size of the network grows (each node can become a CN for an item it

requests if not cached elsewhere in the network), and thus is more suitable to dynamic MANETs than a push-based alternative since the server does not need to be aware of CN disconnections. Client data cache invalidation mechanism is also more suitable when data requests are database queries associated with tables and attributes. In a push-based approach, the server would have to map a cached query to all of its data sources (table attributes) and execute this query proactively, whenever any of the sources is updated. Moreover, client data cache invalidation mechanism adapts the TTL values to provide higher consistency levels by having each CN estimate the inter update interval and try to predict the time for the next update and sets it as the items expiry time. It also estimates the inter-request interval for each data item to predict its next request time, and then pre-fetches items that it expects to be requested soon.

4. CONCLUSION

A Client data cache invalidation mechanism is presented for MANETs that use a special node to cache the queries and the address of the nodes that store the responses to these queries. The proposed mechanism makes use of pull-based algorithm that pulls the data from server and stores it in the client cache node. Also, it relies on estimating the inter update intervals of data items to set their expiry time. It makes use of piggybacking and pre-fetching to increase the accuracy of its estimation to reduce both traffic and query delays.

For future work, first a more sophisticated TTL algorithms needs to be investigated to replace the running average formula. Then extending the preliminary work to secure the data fetched from server as well as delivered to client from cache node by applying cryptography algorithms.

5. REFERENCES

[1] Kassem Fawaz and Hassan Artail, "DCIM: Distributed Cache Invalidation Method for Maintaining Cache Consistency in Wireless Mobile Networks", *IEEE Transactions on Mobile Computing*, vol. 12, no. 4, April 2013.

[2] Priyanka Goyal, Vinti Parmer and Rahul Rishi, "MANET: Vulnerabilities, Challenges, Attacks, Application", *IJCEM International Journal of Computational Engineering & Management*, vol. 11, January 2011.

[3] Pravin Ghosekar, Girish Katkar and Dr. Pradip Ghorpade," Mobile Ad Hoc Networking: Imperatives and Challenges", *IJCA Special Issue on "Mobile Ad-hoc Networks" MANETs*, 2010.

[4] D. Barbara and T. Imielinski, "Sleepers and Workaholics: Caching Strategies for Mobile Environments," *Proc. ACM SIGMOD*, pp. 1- 12, May 1994.

[5] M. Denko and J. Tian, "Cooperative Caching with Adaptive Prefetching in Mobile Ad Hoc Networks," *Proc. IEEE Int'l Conf. Wireless an Mobile Computing, Networking and Comm. (WiMob '06),* pp. 38-44, June 2006.

[6] 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.

[7] G. Cao, "A Scalable Low-Latency Cache Invalidation Strategy for Mobile Environments," *IEEE Trans. Knowledge and Data Eng.*, vol. 15, no. 5, pp. 1251-1265, Sept./Oct. 2003.

[8] J. Jing, A. Elmagarmid, A. Helal, and R. Alonso, "Bit-Sequences: An Adaptive Cache Invalidation Method in Mobile Client/Server Environments," *Mobile Networks and Applications*, vol. 2, pp. 115-127, 1997.

[9] Q. Hu and D. Lee, "Cache Algorithms Based on Adaptive Invalidation Reports for Mobile Environments," *Cluster Computing,* vol. 1, pp. 39-50, 1998.

[10] K.S. Khurana, S. Gupta, and P. Srimani, "A Scheme to Manage Cache Consistency in a Distributed Mobile Wireless Environment," *IEEE Trans. Parallel and Distributed Systems*, vol. 12, no. 7, pp. 686-700, 2001

[11] S. Lim, W.C. Lee, G. Cao, and C. Das, "Cache Invalidation Strategies for Internet-Based Mobile Ad Hoc Networks," *Computer Comm.*, vol. 30, pp. 1854-1869, 2007.

[12] W. Li, E. Chan, D. Chen, and S. Lu, "Maintaining Probabilistic Consistency for Frequently Offline Devices in Mobile Ad Hoc Networks," *Proc. IEEE 29th Int'l Conf. Distributed Computing Systems*, pp. 215-222, 2009.

[13] P. Cao and C. Liu, "Maintaining Strong Cache Consistency in the World-Wide Web," *IEEE Trans. Computers*, vol. 47, no. 4, pp. 445-457, Apr. 1998.

[14] K. Mershad and H. Artail, "SSUM: Smart Server Update Mechanism for Maintaining Cache Consistency in Mobile Environments," *IEEE Trans. Mobile Computing*, vol. 9, no. 6, pp. 778-795, June 2010.

[15] H. Artail, H. Safa, K. Mershad, Z. Abou-Atme, and N. Sulieman, "COACS: A Cooperative and Adaptive Caching System for MANETS," *IEEE Trans. Mobile Computing*, vol. 7, no. 8, pp. 961- 977, Aug. 2008.