MobiCom Poster Abstract: Mitigating the Gateway Bottleneck via Transparent Cooperative Caching in Wireless Mesh Networks

Saumitra M. Das *smdas@purdue.edu*
Himabindu Pucha *hpucha@purdue.edu*
Y. Charlie Hu *ychu@purdue.edu*
School of ECE and Center for Wireless Systems and Applications, Purdue University, W. Lafayette, IN, USA

I. Motivation and Key Idea

Wireless mesh networks (WMNs) are characterized by mesh routers connected by wireless links to each other and to a few gateway nodes. The most significant application of such networks is to provide broadband Internet access to static or mobile hosts in areas where wired infrastructure is difficult or economically infeasible to deploy. Since most WMN traffic flows through a few gateway nodes, this can cause significant congestion around the gateway.

Previous studies have shown that significant locality exists in Internet accesses from a given population of clients. Web caching has been proposed and extensively studied to exploit such locality in reducing the Internet traffic and the client-perceived access latency. We anticipate that similar locality will exist in WMN traffic. We propose *MeshCache*, a transparent cooperative caching system that exploits locality in WMN traffic to mitigate the gateway bottleneck effect. Unlike transparent web caching in the Internet where a web cache is attached to the gateway of an organization, MeshCache leverages the fact that a WMN typically spans a small geographic area and hence mesh routers are easily over-provisioned with CPU, memory, and disk storage, and extends the individual wireless mesh routers in a WMN with built-in content caching functionality. It then performs *cooperative caching* among the wireless mesh routers.

Cooperative caching among mesh routers allows clients to fetch cached data from routers within the WMN. This has several benefits: (1) It spreads the load in the network and hence alleviates the congestion around the gateway. (2) It promotes local communication between clients and nearby routers (that have a cached copy) which improves the capacity of WMNs [1]. (3) It also improves client throughput by allowing content to be fetched from nodes closer (in network hops) than the gateway as well as by providing clients the choice of choosing the *best* cached copy based on link-quality routing metrics (e.g. [2]).

II. Architecture and Algorithms

We explore two architectural design choices for MeshCache (A2 and A3 below), and compare it to a third one (A1), which is similar to the typical way that a web cache is deployed in the Internet. For A2 and A3 we also develop and evaluate 3 cache selection protocols.

A1: A web proxy cache is connected to the WMN gateway node similar to how a web cache is attached to a gateway router in the Internet. This proxy cache transparently hijacks clients' content requests to exploit the locality within the WMN client population.

A2: Each mesh router acts as a cache using expandable storage devices. When a client issues a request for a data object, its access mesh router transparently hijacks the request and searches for the object in its local cache. On a hit, the object is simply served from the access mesh router. On a miss, the access mesh router searches for a cached copy of the data object in other mesh routers including the gateway. If a copy is found, the access mesh router fetches the copy, otherwise it obtains the object from the origin server via a proxy cache at the gateway (as in A1).

A3: In addition to A2, when the data object is fetched either from a gateway or another mesh router along a multi-hop route, the object is cached at each mesh router along the route. This increases the availability of the data object for future requests without explicit replication. To enable such *hop-by-hop caching*, we use *per-hop transport* that breaks a single end-to-end transport connection, e.g. S to D using route S-A-B-D, into multiple single-hop transport connections along the route, e.g. S-A, A-B, B-D, and pipelines data over these sub-connections. This enables the data object to be cached at A and B in addition to S. The architectures A2 and A3 require a cache selection protocol to locate a cached copy of content and/or choose among multiple cached copies.

To this end, we designed and compared three cache selection protocols for each architecture A2 and A3 in MeshCache. The first protocol THCP (Tree-based Hierarchy Cache Protocol) uses a hierarchy of caches, BCP (Broadcast Cache Protocol) does a TTL-limited flood to locate content, and GHCP (Geographic Hash Cache Protocol) hashes the content to a peer mesh router. We also implemented per-hop variant of the above protocols where a connection to the selected mesh router is made using per-hop transport which increases the content availability by caching the content on the routers that *it is passing through*.

III. Key Performance Indicators

Does locality exist? A fundamental question that determines the potential performance benefits of MeshCache is whether there exists significant locality in Internet-bound traffic in a WMN, given the small client population served by each gateway router. We can approximate the client population served by a gateway node of a WMN with that seen by the gateway proxy cache of a small organization. For such an approximation, we analyzed real web proxy traces collected in the week of October 19, 2005 by `www.ircache.net` containing 2.7 million requests from 10 proxies originating from 1151 unique clients to 81,289 servers. The number of clients in each trace ranges from 80 to 160 nodes which is a potential target size for a WMN with a single gateway. For example, a recent work [3] shows that 114 users can potentially be supported with a 21 mesh router WMN.

We studied the locality and working set size of the cacheable content in each trace. The locality in the access patterns of such a small set of clients is encouraging as the average hit rate is around 37% with a maximum of 46%. This suggests that a significant fraction of requests can be fetched from peer mesh routers if caching is enabled. We also found that the hit rate increases gradually from 35% to 45% as the client population grows from 50 to 500. In summary, there is significant locality in smaller client populations such as those in WMNs. Another property of the Internet access traffic which determines the feasibility of caching is the working set size of the traffic. We found that the footprint of a day's worth of traffic to be 1.3 GB on average with a maximum of 2.6 GB. This footprint can be accommodated in expandable storage devices currently available.

Which architecture/algorithm work the best? We vigorously evaluated the performance of MeshCache using simulations in a detailed simulator Glomosim and testbed experiments conducted on a subset of our 32-node multi-radio wireless network testbed [4] with an Internet-like traffic pattern. Our evaluation results shows that A3 (cooperative caching with hop-by-hop caching) outperforms the other architectures in reducing the gateway load and improving the client throughput, irrespective of the cache selection protocol used. Additionally, we found that the per-hop transport for hop-by-hop caching in A3 provides increased content availability without adversely affecting transport throughput or network overhead compared to end-to-end transport. In fact, the throughput of per-hop transport is better than end-to-end transport under fading channels with lossy links. Since these conditions are typically true in wireless networks, we expect per-hop transport to generally perform better than or comparable to end-to-end transport, thereby *improving caching without a performance penalty*. Note that our simulations capture the delay of traversing up and down the TCP/IP stack when simulating per-hop transport.

Further, for the best performing architecture A3, we found that the cache selection based on *broadcast-based limited TTL flooding* (per-hop BCP) is the best strategy to alleviate the gateway bottleneck and obtain throughput improvement. Particularly, our simulations showed that: (1) A2 increased client throughput by up to 50% and reduced the gateway load by up to 35% compared to A1, (2) A3 increased throughput by up to 52% and reduced the gateway load by up to 66% compared to A2, and (3) the best cache selection strategy for A3 is per-hop BCP which reduces the gateway load by up to 20% compared to the next best strategy. Our testbed experiments also show that MeshCache can improve the overall throughput and reduce the network load significantly. Specifically, measurement results from a deployed implementation of MeshCache on the MAP mesh network testbed [4] show that the number of transfers that achieve a throughput greater than 1 Mbps is increased from 20% in A1 to 60% in A3 while the load at the gateway is reduced by 38% in A3 compared to A1.

IV. Conclusions and Future Work

Our MeshCache system alleviates the gateway congestion in WMNs while improving client throughput. We are currently working on exploiting the MeshCache content cache for enabling applications other than Internet access such as community media sharing and software distribution. A detailed document describing this work is available [5].

References

[1] J. Li, C. Blake, D. S. D. Couto, H. I. Lee, and R. Morris. Capacity of ad hoc wireless networks. In *Proc. of ACM MobiCom*, March 2001.

[2] D. S. J. D. Couto, D. Aguayo, J. C. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proc. of ACM MobiCom*, 2003.

[3] J. Eriksson, S. Agarwal, V. Bahl, and J. Padhye. A feasibility study of mesh networks for an all-wireless office. In *Proc. of ACM MobiSys*, 2006.

[4] Mesh@Purdue. `https://engineering.purdue.edu/MESH`.

[5] S.M.Das, H.Pucha, and Y.C.Hu. Mitigating the Gateway Bottleneck via Transparent Cooperative Caching in Wireless Mesh Networks. Technical report, TR-ECE-06-07, Purdue University, March 30 2006. `http://shay.ecn.purdue.edu/~dsnl/mcache.pdf`.