HTTP Traffic Offload in Cellular Networks via WLAN Mesh Enabled Device to Device Communication and Distributed Caching

Chris Drechsler, Marco Porsch, Gerd Windisch

HTTP Traffic Offload in Cellular Networks via WLAN Mesh enabled Device to Device Communication and Distributed Caching

Chris Drechsler, Marco Porsch, Gerd Windisch

Chemnitz University of Technology
Chemnitz, Germany
{chris.drechsler|marco.porsch|gerd.windisch}@etit.tu-chemnitz

Abstract. Mobile network operators are currently observing a tremendous increase of mobile data traffic due to the increased usage of smartphones or other Internet-enabled devices (e.g. tablets, laptops). A natural solution for reducing the transmission costs and for improving the QoE is caching of frequently requested content. In this contribution we propose a distributed caching architecture where the clients perform the caching of frequently requested HTTP content by allowing a partial use of the user equipments memory for local caching. Furthermore we engage clients to exchange content via WLAN mesh enabled device-to-device communication in order to offload traffic from the 3GPP radio access network.

Keywords: HTTP caching, device-to-device communication, cooperative caching

1 Introduction

Internet usage has significantly evolved in the last few years and the growing data traffic contributes substantially to risings costs for network operators. Especially cellular network operators observe a tremendous increase of mobile data traffic due to increased usage of smartphones or other Internet-enabled devices (e.g. tablets, laptops). According to a recent study of Cisco [2] the global mobile data traffic grew by 70 % in 2012. To cope with the increasing data traffic in mobile networks and to reduce transmission costs, caching of frequently requested content within operator networks is a natural solution. Recent studies have shown that HTTP traffic accounts for up to 82 % of the average downstream traffic in mobile networks [5] and that the potential of HTTP caching is quite high. Between 28 % - 68 % of all bytes transferred via HTTP are basically cacheable [6], [3].

In this context we investigate the question of applying caching in mobile networks. An obvious approach would be to deploy classic HTTP cache systems like Squid [1] at different locations within the core network of the mobile operator, e.g. at S-/P-GW sites. In contrast to that we propose a distributed caching approach where the clients perform the caching of frequently requested HTTP

content. By allowing a partial use of the user equipments (UEs) memory for local caching (which is free of charge from the operators perspective), clients are engaged to share their already downloaded HTTP content with other clients. Besides the 3GPP air interface for default Internet access, we assume that the clients in parallel participate in an open WLAN mesh network via their WLAN interface. Thus, by device-to-device communication over the WLAN mesh network they can directly support each other in retrieving frequently requested HTTP content.

The paper is organized as follows: Section 2 provides an overview of related work in the context of caching in cellular networks close to UEs (e.g. at eNodeB sites) and of device-to-device communication. In Section 3 the new distributed cooperative caching approach is described and detailed information about its operation is provided. Section 4 concludes the paper.

2 Related Work

Ahlehagh et al. introduced in [4] new ideas for caching video content at base stations. They propose to provide a large number of micro-caches (with up to 50 GByte capacity) basically one micro-cache for each base station. Together with new replacement strategies they demonstrate the feasibility and effectiveness of caching at the edge of the RAN. However contrary to our work they do not consider a distributed and cooperative caching approach with caching performed in the UEs.

Most contributions related to device-to-device communication in cellular networks (like, e.g. [7]) contrary to our work assume that 3GPP radio resources are used for the device-to-device communication. Thus the main research question is on how to opti-mally allocate radio resources for device to device communication and how to minimize interference.

Other contributions consider traffic offload in mobile networks via WLAN hotspots [8]. In contrast to our work the focus is on offloading delay tolerant traffic to the Internet. WLAN is used in infrastructure mode and not for device to device com-munication.

3 Distributed cooperative HTTP caching

Our distributed and cooperative caching approach is based on three main concepts:

- HTTP header field extension
- distributed caching architecture
- distributed cache operation

In the following, these main concepts are explained in more detail.

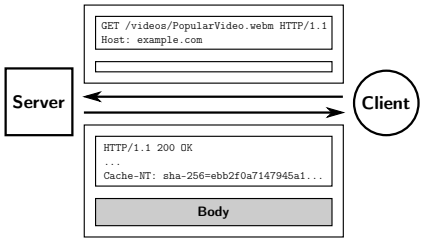

Fig. 1. HTTP header field extension

```
<?xml version="1.0" encoding="UTF-8"?>
<ClientCacheIndex>
    <CacheEntry>
        <SHA256ID>e70d4...92dbb<\SHA256ID>
    <\CacheEntry>
    <CacheEntry>
        <SHA256ID>d46f6...dd3a2<\SHA256ID>
    <\CacheEntry>
...
</ClientCacheIndex>
```

Fig. 2. ClientCacheIndex.xml

3.1 HTTP Header Field Extension

One reason for the partly unused caching potential is that todays HTTP cache systems like Squid [1] mainly rely on the URL to detect duplicate transfers. But mechanisms like server load balancing and the placement of content on several servers within CDNs often lead to different URLs for one specific resource (URL aliasing). Another reason is the personalization of HTTP messages (e.g. via cookies or parameters in the query string).

To overcome this issues we propose to use a hash key for identifying the HTTP content more precisely than with URLs. The hash value is computed over the specific representation of a requested resource and is added to the header of the HTTP response messages (see figure 1).

3.2 Distributed Caching Architecture

The distributed caching architecture consists of two main entities (see figure 3):

- **Origin HTTP server:** The origin HTTP server is located in the domain of the content producer and acts like a normal HTTP server but with one difference: it adds the above explained header attribute with the hash value of the requested resource to the header of the transferred HTTP message (for details see chapter 3.1).
- **Sharing Clients:** Sharing clients are the UEs in the domain of the mobile operator. On the one hand they are connected to the Internet via the 3GPP air interface and on the other hand they participate in an open WLAN mesh network via their Wi-Fi interface in order to directly share resources with nearby UEs. The sharing clients consist of the following three sub-entities:
  - **User agent:** Sharing clients can act like normal clients by retrieving web content with an ordinary user agent (web browser) like e.g. Firefox. The only change in our architecture is that the user agent forwards the HTTP traffic to the smart proxy (see below).
  - **HTTP server:** The sharing clients are running a normal HTTP server which acts like a simple file server with a flat hierarchy. Shared resources can be identified and accessed by their hash value (see chapter 3.1). To announce which resources are shared, the client offers an index. In

our approach a simple xml file (called ClientCacheIndex.xml) is made available on the HTTP server of each sharing client (see figure 2).

- **Smart proxy:** The smart proxy is located in the data path between the origin server and the user agent (web browser), see figure 3. It serves as a central element and manages the data exchange of web content. By default the 3GPP interface is used. In case that at least one sharing client participating in the WLAN mesh network has a copy of the requested content, the WLAN interface is used. For a detailed explanation of the cache operation see chapter 3.3. The smart proxy fulfills the following two basic functions:
  * Indexing: The smart proxy builds an index of all resources that are shared by nearby sharing clients. For that it periodically (e.g. every 5-10 minutes) scans the mesh network for available sharing clients/UEs and queries each of them for the index of its shared HTTP resources.
  * Analyzing HTTP traffic: To enable the distributed cache operation the smart proxy analyzes all incoming HTTP traffic on the 3GPP interface wrt. the HTTP header attributes (that includes the hash value of the transferred content). In case a copy of the requested resource is available by at least one nearby sharing client/UE the smart proxy intercepts the HTTP transfer from the origin HTTP server and retrieves the content from the nearby sharing client.

3.3 Distributed Cache Operation

To explain the distributed cache operation in more detail we provide the following example which is based on the scenario depicted in figure 3.

The user agent of client A sends a HTTP request message (method GET) for a desired resource to its smart proxy (1), which forwards it directly to the origin HTTP server (2). The origin HTTP server answers with a HTTP response message (3) and adds the header attribute with the computed hash value of the requested resource to the header of the transferred HTTP message. The smart proxy of client A analyzes the incoming HTTP response message and looks for the header attribute (4). In the example it is assumed that the smart proxy recognizes that a nearby sharing client B in the WLAN mesh network has an object with the same hash value. Therefore the smart proxy sends a HTTP request message (method GET) for the resource to the HTTP server of client B (5) and the client returns a HTTP response message with the resource contained in the body (6). Thus, the smart proxy receives the resource from both, the origin HTTP server via the 3GPP interface and the client B via device to device communication over the WLAN mesh network. It concatenates the HTTP header received by the origin HTTP server with the HTTP body received by client B and forwards the message to the user agent of client A (7). As soon as the transfer is successful established the smart proxy aborts the connection to the origin HTTP server (8).

There are frequent concerns regarding the sharing of copyright protected content in the Internet. With our caching architecture this concerns are repealed.

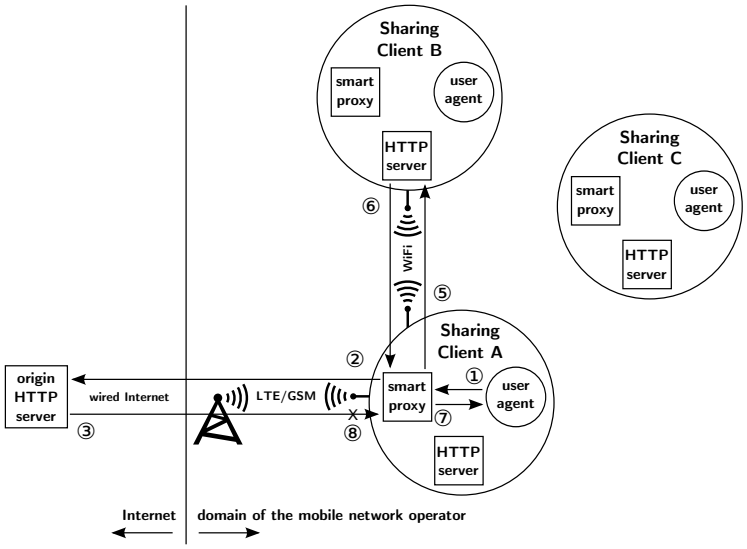

Fig. 3. HTTP data flow in the distributed caching architecture (example)

As explained above, the origin server is always requested. If it responds with a 2xx HTTP response code (and if the header field extension is applied) then the smart proxy can intercept the transmission and the content might be retrieved from a nearby sharing client. On the other hand this means that no content can be retrieved from sharing clients if the content isnt available on the origin server.

4 Conclusion

In this contribution we present a distributed caching approach where the clients (UEs) perform the caching and support each other in retrieving frequently requested HTTP content via device-to-device communication in a WLAN mesh network. We already implemented the concepts of our approach in a testbed where notebooks are used instead of UEs. Currently we are replacing the notebooks with real smartphones and are porting the software to Android OS. For future work we plan to perform a comprehensive performance evaluation of our approach

References

[1] Homepage squid project. `http://www.squid-cache.org/`
[2] Cisco visual networking index: Global mobile data traffic forecast update, 2012-2017 (6 Feb 2013), `http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.html`

[3] Ager, B., Schneider, F., Kim, J., Feldmann, A.: Revisiting cacheability in times of user generated content. In: INFOCOM IEEE Conference on Computer Communications Workshops , 2010. pp. 1–6 (march 2010)
[4] Ahlehagh, H., Dey, S.: Video caching in radio access network: Impact on delay and capacity. In: Wireless Communications and Networking Conference (WCNC), 2012 IEEE. pp. 2276–2281 (2012)
[5] Erman, J., Gerber, A., Hajiaghayi, M., Pei, D., Sen, S., Spatscheck, O.: To cache or not to cache: The 3g case. Internet Computing, IEEE 15(2), 27–34 (2011)
[6] Erman, J., Gerber, A., Hajiaghayi, M.T., Pei, D., Spatscheck, O.: Network-aware forward caching. In: Proceedings of the 18th international conference on World wide web. pp. 291–300. WWW '09, ACM, New York, NY, USA (2009)
[7] Fodor, G., Dahlman, E., Mildh, G., Parkvall, S., Reider, N., Miklos, G., Turanyi, Z.: Design aspects of network assisted device-to-device communications. Communications Magazine, IEEE 50(3), 170–177 (2012)
[8] Ristanovic, N., Le Boudec, J.Y., Chaintreau, A., Erramilli, V.: Energy efficient offloading of 3g networks. In: 8th International Conference on Mobile Adhoc and Sensor Systems (MASS), 2011 IEEE. pp. 202–211 (2011)