Caché: Caching Location-Enhanced Content to Improve User Privacy

Shahriyar Amini
Carnegie Mellon University
Pittsburgh, PA
shahriyar@cmu.edu

I. Problem and Motivation

In recent years, location-aware devices, such as GPS-enabled mobile phones, have gained mainstream popularity. This trend has led to a rapid increase in location-based services beyond just navigation [5, 37]. Examples of such location-based services include support for finding gas stations[1] and stores[2] with the lowest prices, finding friends, and notifying friends when you arrive at a location. Other examples include location-based games[3] as well as location-enhanced micro-blogging.

A key challenge to widespread adoption of location-based services, however, is privacy [22]. One problem here is the perception of privacy: people have expressed many concerns about being tracked by friends and by third parties [13], as noted in numerous interviews [18, 21], essays [11, 35, 38], and books [6, 14]. Location privacy concerns also tend to attract negative media coverage [33, 39], further hindering the spread of location-based services. Another problem here is actual privacy: end-users may be unaware of the privacy implications of location-based technologies [3, 4], and end up unintentionally sharing more information than they realized.

To address the aforementioned problems, we present Caché, a generalizable approach for a class of location-based services that enables users to enjoy the benefits of those services while minimizing the associated privacy concerns. Caché takes a well-explored idea from systems, namely caching, and applies it in the context of privacy. Caché has two core ideas: (1) location-enhanced content can be periodically pre-fetched in large geographic blocks onto a device before it is actually needed, for areas that a person will likely be in, and (2) the content can be accessed locally on a device when it is actually needed, without relying on any networked services outside of the device. Thus, rather than sharing current location on each request for information, the user only needs to share general geographic region hours, days, or even weeks before the desired content is needed.

There are four steps in using Caché. Let's consider a scenario where a restaurant finder application is developed to use Caché. First, at design time, the developer provides some hints as how to download the content (e.g. URL, content update interval). Second, the user installs the Caché-enabled application and selects for what regions restaurant POIs should be downloaded. Third, Caché downloads and updates the content based on the developer specified update rate, at a favorable time (e.g. device is powered and a WiFi connection is present). Finally, when the application requires content, it retrieves it from Caché, rather than make a live query.

Assuming that the user's current location is determined locally on the device (as is the case with GPS, PlaceLab [19, 26], POLS [30], and SkyHook's autonomous mode), the user can still make use of location-enhanced content, while content providers that offer maps, restaurant guides, bus schedules, and other location-enhanced content are only aware of the user's general area of interest. Because the user's actual location is not revealed to external entities at the time of use, user privacy is maintained.

The full analysis, design, evaluation, and results of Caché have been accepted to MobiSys 2011 [2]. Readers interested in implementation specifics and additional detail should refer to the full technical report version [1].

II. Background and Related Work

In this section, we present the background and related work to the Caché approach. We start with content caching and proceed to discuss location privacy.

Content Caching: Caching content has been a well-explored topic for mobile computing, primarily focused on performance or disconnected or weakly connected devices. For example, the Bayou architecture is designed from the ground up to support mobile computing applications [9], while the Coda file system is designed to provide support for weakly con-

[1] GasBag, http://www.jam-code.com
[2] ShopSavvy, http://www.biggu.com
[3] JOYity, http://www.androidapps.com/t/joyity

nected operation [23, 31]. Our work differs from this previous work in that we apply caching for location privacy, examining the tradeoffs of such an approach in today's Internet architecture.

Location Privacy: Many past projects have explored balancing the tradeoffs involved in providing useful functionality while offering privacy protection to end-users. Recent work by Brush et al. looks at if and how users understand the effectiveness of various privacy preserving techniques and how much users value their location data in monetary terms [7].

Duckham and Kulik [12] sketch out four major themes for location privacy, namely regulation, privacy policies, anonymity, and obfuscation. Krumm [24] offers a survey of computational approaches to location privacy that examines inference techniques as well as countermeasures. Issues surrounding regulation and privacy policies are beyond the scope of this paper. Instead, we will focus on the other two areas.

Anonymity: A common theme in much of the work on anonymity has been relying on a trusted third party that acts as a proxy between the client and the location-based service. One metric that has been developed is k-anonymity [15, 34]. There are many examples of work in location privacy using k-anonymity. Perhaps the most relevant here are spatial and temporal cloaking [16], Mix Zones [17, 4], New Casper [28], and (to some extent) CacheCloak [27]. Our work here differs in that Caché does not focus on anonymity. Instead, Caché offers a different model for accessing location-enhanced content, one that relies on pre-fetching and disconnected operation.

Anonymizing networks may also be used to preserve user privacy. Tor [10] is one practical implementation of a low-latency anonymizing network. However, the performance issues associated with Tor, specially in mobile applications, compromise application usability. Further, Tor does not protect against services that require authorization. With respect to anonymity approaches, Caché does not require any third party, nor does it require a critical mass of users to be effective.

Anonymity can also be established with encryption techniques. Private Information Retrieval (PIR) [8, 25, 29] allows clients to make queries to a server in a way that the server cannot distinguish which memory address was read. The approach is interesting, however, support would need to be implemented both on client and server side, and use of cryptography would introduce additional overhead to the system.

Obfuscation: Many techniques have been developed, including adding noise, quantizing locations (essentially putting locations into buckets or aligned onto a grid), and adding false locations. SybilQuery is a client-side tool that creates many different queries to the server to obfuscate the user's actual path [32]. Caché relies on obfuscation in that it only shares with content providers that the user is in a geographic region, e.g. a neighborhood or a city. However, Caché uses obfuscation in a different way than past work, in that after retrieving content for a region, it processes and filters content locally on the user's mobile device. Thus, content providers only know that the user is in a large region rather than obfuscated trails or locations.

Privacy-Enhancing Systems: Finally, we discuss two systems that are perhaps the closest to Caché in terms of goals. Confab is a framework with the purpose of providing support for building ubiquitous computing application with privacy enhancing mechanisms [20]. Caché is a logical extension of this past work as it vastly expands the kind of data types available for application development. Caché also provides deeper analysis of the tradeoffs involved. Another approach for hiding the user's location by surrounding it with other users' paths is CacheCloak [27]. CacheCloak caches all previous requests and services queries from the cache first. If data is not available, it makes a live query, disguising the user's current location by requesting data along the predicted path, extended until the path intersects with other paths. CacheCloak shares similar design ideas to our approach. However, predicting mobility is not always necessary. We show that pre-fetching is feasible for a substantial amount of data without hindering system usability.

III. Approach and Uniqueness

In this section, we present an analysis of the feasibility of the Caché approach, continue with the design requirements and specifics of our architecture, and end with why our approach is unique.

III.A. Feasibility Analysis of Caching Content for Privacy

A number of challenges exist when using caching as a solution for privacy, namely data freshness and consistency, storage, and bandwidth. We provide analysis of the technical challenges associated with our model.

Data Consistency: Caché has a simple consistency model, in that it only reads data from the web, and never writes data back. Caché considers data on web sites to be canonical, and data stored on a mobile device to be a soft copy, to be overwritten as needed.

Data Freshness: We analyzed location-enhanced content downloaded daily for a period of five months, namely, weather from Google, events from zvents.com, bus schedules from the Pittsburgh Port Authority, restaurant points of interests from MSN and Yelp, and Google map tiles. Content was downloaded for the entire city of Pittsburgh. The rationale was to assess whether the selected content types could be cached overnight and still provide fresh accurate data when mobile and disconnected.

We initiated downloads in May 2009. We studied each data type with respect to the percentage of data added, removed, and modified daily (See Table 1). The daily percentage of change for map tiles, bus schedules, and Yelp's points of interests are well below 0.20 % with many being 0.00 %. The highest percentage of change for MSN and events are 6.80 % and 11.75 %, respectively. Although the percentage of change for weather forecasts is high, it is usually requested on a city or ZIP code level. Based on our findings for the average daily change in content, we have concluded that it is feasible to download the aforementioned data types for a local mobile device cache to preserve user privacy.

Estimated Storage Requirements: The amount of storage required for the aforementioned content for the entire city of Pittsburgh is less than 20 MB. We interpolated the estimates for New York City based on area, number of points of interests and events. The total storage for New York City with respect to the content mentioned is less than a 100 MB. We also note that there are content types that are not suitable for pre-fetching such as high definition video streams.

Estimated Bandwidth Requirements: Bandwidth is a potential challenge since it could take an unreasonable amount of time to download content. We estimate the time to download content based on a connection speed of 200 kbps, the FCC minimum for high-speed Internet access [36]. Assuming the worst case of refreshing all the content, it would take about 2 hours to complete for New York City. This would be enough time to cache overnight for a complete download.

Data Type	Added	Removed	Modified
Weather	25.00 %	25.00 %	67.26 %
Events	5.28 %	5.35 %	11.75 %
Yelp POI	0.15 %	0.06 %	0.04 %
MSN POI	6.69 %	6.80 %	1.43 %
Bus Schedule	0.00 %	0.00 %	0.15 %
Map Tiles	0.00 %	0.00 %	0.00 %

Table 1: Average daily percentage of change for added, removed, and modified content for various data types. Results are based on downloading the respective content daily for five months starting in late May 2009.

III.B. Design Requirements

We briefly outline the design requirements for Caché in this section. For deployability, it is preferable that a privacy-preserving approach relies only on the mobile device, and minimizes reliance on infrastructure beyond content itself. Service providers might not have incentives to preserve users' privacy, which is the motivation for our work, and further, relying on the mobile device simplifies trust assumptions. Furthermore, we aim to minimize required user interaction. Preferably, a privacy-preserving approach would be completely transparent to the users and application developers, however, this is not a practical approach. As such, we place the burden of managing privacy on application developers, and offer support to simplify the task of making applications privacy sensitive.

Threat Model: Our aim is to minimize the information flow towards location-based service providers, or anyone accessing their logs or observing the traffic on the way. Further, we explicitly desire that the real-time and precise location of Caché users cannot be accurately determined. However, we do not protect against a local eavesdropper such as the network access-point, which can directly observe that the user is downloading traffic related to multiple locations. A detailed applicable threat model for the problem space has been throughly discussed e.g. by Gruteser and Grunwald [16] before. We emphasize that the threat is not only that somebody discovers the location and regular patterns of users movements through LBS or logs of intermediate servers. These logs can also be used to reveal the sender of a message, if the attacker knows that a location belongs to a user, and discovers that the user was in that location at a particular time.

III.C. Caché System Architecture

The Caché architecture is similar to that of a non-transparent Internet proxy requiring application registration. Upon first use, the mobile user specifies regions for which content should be downloaded for the application. At this point, Caché downloads and stores all of the necessary content. Content is updated based on an update rate defined at registration time by the developer. Finally, the application forwards all of its queries for content to Caché instead of the service provider. Caché processes the query and provides the application with the requested content.

Caché relies on space discretization to pre-fetch content for entire continuos regions, e.g. Pittsburgh. Location-enhanced queries generally use latitude and longitude to describe geographical regions. However, it is not possible to download content for every latitude and longitude combination. To address this problem, Caché decomposes a geographic region of interest into a grid of cells. The grid consists of same size rectangular cells. The size of the cells are defined by the developer at the application registration stage. This requires the developer to have a notion of how densely the content is packed in terms of physical geographical space. Caché allows for two grid enhancements to enable more thorough pre-fetching. Using a grid overlay, a shifted version of a grid, Caché allows content to be downloaded such that when a query is located near a corner of an original grid cell, it can be fielded by the center of a cell in the overlay. Using a grid hierarchy, Caché allows for multiple grids to cover the same region, however, in every new iteration of the grid, the size of the cells from the previous grid is quadrupled. The developer specifies whether grid hierarchy and/or overlay should be used, both binary options.

Currently, Caché only allows the pre-fetching of REST requests rather than specific programming language APIs. Popular mobile platforms support REST requests, however, language specific APIs are not universally supported by all platforms. The developer provides Caché with the content request URL with location parameter arguments replaced by Caché supported tags. The tags inform Caché about how location arguments should be provided to sweep through an entire region and pre-fetch content. Finally, the developer provides the update rate for the content in terms of days and a schedule priority for cases when several applications have content updates on the same day.

After application installation, during the initial use of the application, the user provides Caché with regions for which content should be downloaded, based on an anchor and a radius. The anchor point is the center of geographical region for which content should be downloaded, e.g. an address, ZIP code, or a city. The radius is specified in miles. Once regions have been specified, Caché pre-fetches the content. It also updates the content based on the update rate provided by the developer. When content queries are made, it offers the content of the smallest cell in the grid(s) that covers the entire query region. If there is no content available, Caché makes a live request.

III.D. Uniqueness of Approach

Our approach is unique in that it uses pre-fetching and caching content in a new light. Specifically, instead of relying on pre-fetching and caching to improve performance or to enable content use on disconnected or weakly connected devices, Caché uses them to enhance user privacy. Of course, this does not diminish the benefits of pre-fetching and caching. Instead, it makes Caché appealing for developers as it boost application performance while reducing the burden to make the application privacy friendly. In other words, unlike most approaches to location privacy, Caché provides developers with a strong incentive to increase privacy because pre-fetching and caching improve performance and energy efficiency, and enable disconnected operation. In a practical sense, our approach has a high chance of being adopted versus complex privacy preserving solutions that require additional infrastructure or developers with expert knowledge of preserving user privacy.

IV. Results and Contributions

We evaluated Caché using two separate methods. One method analyzed the content hit rate based on two mobility trace datasets. We report the results from only one of the datasets here[4]. The hit rate evaluation consisted of estimating the locations of a person's home and work, and then downloading all of the content within a certain radius. We measured the hit rate as the number of actual locations users visited that fell within the two regions centered at home and at work. The other evaluation method focused on modifying applications to use Caché. We modified two open-

[4] http://locaccino.org/

source applications and also a restaurant finder application, which was developed by the authors. We implemented Caché as an Android service with an interface that may be added to and used by any Android project.

IV.A. Results

Mobility traces from the top 20 most active users were used for evaluation, consisting of the latitude and longitude, sampled at approximately five minute intervals. The data focused on the period from mid December 2008 up until late October 2009. Based on 5, 10, and 15 mile radii, Caché provided approximately 86% hit rate. The improvements in increasing the pre-fetching radius were not substantial. The other dataset, not explored in detail here, provided a 78%, 83%, and 86% hit rate for the aforementioned radii, respectively.

Three open-source applications were modified to use the Android Caché service. Mixare is an augmented reality engine which presents nearby content to the user. Panoramio shows nearby pictures that have been uploaded by other users. Restaurant Request is an open source application written by the authors to request nearby restaurant points of interest from Yelp. The three applications have 4692, 1268, and 411 source lines of code, respectively. There were 18 (0.4%), 18 (1.4%), and 12 (2.9%) source lines of code added, and 4 (0.1%), 7 (0.55%), and (1.9%) source lines of code removed, respectively. The majority of the change involved routing the HTTP request to Caché rather than to the Android stack. Considering the small change in application source code, Caché does not pose a large burden on the developer to improve privacy.

Because of the inherent advantages of pre-fetching and caching, the application experiences improved performance and functionality, especially while disconnected or weakly connected to the Internet. The high percentage of cache hit rate, the benefits of caching, and the small additional effort required by developers make Caché an appealing solution for preserving user privacy while improving application performance.

IV.B. Contributions

Our research efforts make several significant contributions to the field of computer science. Perhaps the highlight of our findings is that improving user privacy does not always require complex solutions and additional infrastructure. Because of space limitations, we presented only an overview of the analysis, design, evaluation, and results of Caché here. However, our work is accessible as a full accepted submission to MobiSys 2011 [2] and also as a comprehensive technical report [1].

Specifically, Caché makes the following contributions to the wider computer science community. It provides a feasibility analysis of caching for privacy, including a taxonomy of location-based data types, and a discussion of tradeoffs with respect to freshness of data, storage, and bandwidth requirements. It presents a system architecture that through pre-fetching enables the use of location-enhanced content while also supporting user privacy. Further, it offers a reference implementation of the approach. Caché also offers a performance analysis that demonstrates the benefits of caching, specifically, the increase in privacy with respect to increase in bandwidth and storage usage, evaluated through the use of two real-world mobility trace datasets. Finally, Caché describes our experiences using the approach to improve privacy in three open-source Android applications, showing the minimal effort required by developers to make their applications privacy conscience.

References

[1] S. Amini, J. Lindqvist, J. I. Hong, J. Lin, E. Toch, and N. Sadeh. Caché: Caching location-enhanced content to improve user privacy [extended]. Technical Report CMU-CyLab-10-019, CMU Cylab, 2010.

[2] S. Amini, J. Lindqvist, J. I. Hong, J. Lin, E. Toch, and N. Sadeh. Caché: Caching location-enhanced content to improve user privacy. In *Proc. of MobiSys (To Appear)*, 2011.

[3] L. Barkhuus and A. Dey. Location-Based Services for Mobile Telephony: a study of user's privacy concerns. In *Proc. of Interact*, July 2003.

[4] A. R. Beresford and F. Stajano. Location Privacy in Pervasive Computing. *IEEE Pervasive Computing*, 2(1), 2003.

[5] E. Biba. Inside the GPS Revolution: 10 Applications That Make the Most of Location. *Wired Magazine*, 17(2), 2009.

[6] D. Brin. *The Transparent Society*. Perseus Books, 1998.

[7] A. B. Brush, J. Krumm, and J. Scott. Exploring end user preferences for location obfuscation, location-based services, and the value of location. In *Proc. of UBICOMP*, 2010.

[8] B. Chor, O. Goldreich, E. Kushilevitz, and M. Su-

dan. Private Information Retrieval. In *Proc. of FOCS*, 1995.

[9] A. Demers, K. Petersen, M. Spreitzer, D. Terry, M. Theimer, and B. Welch. The Bayou Architecture: Support for Data Sharing Among Mobile Users. In *Proc. of WMCSA*, Dec. 1994.

[10] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *In Proceedings of the 13 th Usenix Security Symposium*, 2004.

[11] S. Doheny-Farina. The Last Link: Default = Offline, Or Why Ubicomp Scares Me. *Computer-mediated Communication*, 1(6), 1994.

[12] M. Duckham and L. Kulik. *Location privacy and location-aware computing*, chapter 3. CRC Press, 2006.

[13] N. Eagle. Behavioral Inference across Cultures: Using Telephones as a Cultural Lens. *IEEE Intelligent Systems*, 23(4), 2008.

[14] S. Garfinkel. *Database Nation: The Death of Privacy in the* 21^{st} *Century*. O'Reilly & Associates, 2001.

[15] B. Gedik and L. Liu. Protecting location privacy with personalized k-anonymity: Architecture and algorithms. *IEEE TMC*, 2007.

[16] M. Gruteser and D. Grunwald. Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking. In *Proc. of MobiSys*, 2003.

[17] M. Gruteser and B. Hoh. On the anonymity of periodic location samples. In *Proc. of SPC*, 2005.

[18] R. H. R. Harper. Why Do People Wear Active Badges? Technical Report EPC-1993-120, EuroPARC, 1993.

[19] J. Hong, G. Borriello, J. Landay, D. Mcdonald, B. Schilit, and D. Tygar. Privacy and Security in the Location-enhanced World Wide Web. In *Proc. of UBICOMP*, 2003.

[20] J. I. Hong and J. A. Landay. An architecture for privacy-sensitive ubiquitous computing. In *Proc. of MobiSys*, 2004.

[21] E. Kaasinen. User needs for location-aware mobile services. *Personal Ubiquitous Comput.*, 7(1), 2003.

[22] W. Karim. Privacy Implications of Personal Locators: Why You Should Think Twice before Voluntarily Availing Yourself to GPS Monitoring. *Washington University Journal of Law and Policy*, 14(485), 2004.

[23] J. J. Kistler and M. Satyanarayanan. Disconnected operation in the Coda File System. *ACM Trans. Comput. Syst.*, 10(1), 1992.

[24] J. Krumm. A survey of computational location privacy. *Personal Ubiquitous Comput.*, 2008.

[25] E. Kushilevitz and R. Ostrovsky. Replication is not needed: single database, computationally-private information retrieval. In *Proc. of FOCS*, 1997.

[26] A. LaMarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, T. S. James Scott, J. Howard, J. Hughes, F. Potter, J. Tabert, P. Powledge, G. Borriello, and B. Schilit. Place lab: Device positioning using radio beacons in the wild. In *Proc. of Pervasive*, 2005.

[27] J. Meyerowitz and R. Roy Choudhury. Hiding stars with fireworks: location privacy through camouflage. In *Proc. of MobiCom*, 2009.

[28] M. F. Mokbel, C.-Y. Chow, and W. G. Aref. The new Casper: query processing for location services without compromising privacy. In *Proc. of VLDB*, 2006.

[29] F. Olumofin, P. K. Tysowski, I. Goldberg, and U. Hengartner. Achieving efficient query privacy for location based services. In *Proc. of PETS*, 2010.

[30] Pols. Privacy Observant Location System. *http://pols.sourceforge.net*, 2008.

[31] M. Satyanarayanan, J. J. Kistler, L. B. Mummert, M. R. Ebling, P. Kumar, and Q. Lu. Experience with disconnected operation in a mobile computing environment. In *Proc. of MLCS*, 1993.

[32] P. Shankar, V. Ganapathy, and L. Iftode. Privately querying location-based services with SybilQuery. In *Proc. of Ubicomp*, 2009.

[33] L. Sloane. Orwellian Dream Come True: A Badge That Pinpoints You. *New York Times*, page 14, 1992.

[34] L. Sweeney. k-anonymity: a model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5), 2002.

[35] S. Talbott. The trouble with ubiquitous technology pushers. In *Proc. of CFP*, 2000.

[36] D. Turner. Broadband Reality Check: The FCC ignores America's Digital Divide. *Consumer Union*, 2005.

[37] S. Wang, J. Min, and B. Yi. Location Based Services for Mobiles: Technologies and Standards. In *Proc. of ICC*, 2008.

[38] M. Weiser, R. Gold, and J. S. Brown. The origins of ubiquitous computing research at PARC in the late 1980s. *IBM Syst. J.*, 38(4), 1999.

[39] J. Whalen. You're Not Paranoid: They Really Are Watching You. *Wired Magazine*, 3(3):85–95, 1995.