A distributed infrastructure supporting personalized services for the Mobile Web

Claudia Canali, Michele Colajanni, Riccardo Lancellotti
Department of Information Engineering
University of Modena and Reggio Emilia
{canali.claudia, colajanni, lancellotti.riccardo}@unimo.it

Philip S. Yu
IBM T.J. Watson Research center
psyu@us.ibm.com

Abstract

Personalized services are a key feature for the success of the next generation Web that is accessed by heterogeneous and mobile client devices. The need to provide high performance and to preserve user data privacy opens a novel dimension in the design of infrastructures and request dispatching algorithms to support personalized services for the Mobile Web. Performance issues are typically addressed by distributed architectures consisting of multiple nodes. Personalized services that are often based on sensitive user information may introduce constraints on the service location when the nodes of the distributed architecture do not provide the same level of security. In this paper, we propose an infrastructure and related dispatching algorithms that aim to combine performance and privacy requirements. The proposed scheme may efficiently support personalized services for the Mobile Web especially if compared with existing solutions that separately address performance and privacy issues. Our proposal guarantees that up to the 97% of the requests accessing sensitive user information are assigned to the most secure nodes with limited penalty consequences on the response time.

1 Introduction

Mobile portable devices have already outnumbered traditional desktop computers and will mold the view of future Web-based services. This evolution is even more important since it is combined with the growing amount of personalized services offered to the users. Tailoring Web resources to the user preferences, context, location and to the capabilities of their heterogeneous client devices requires on-the-fly content generation, because a pre-generation of formats for any combination of devices, user needs and contexts is simply unfeasible.

From a computational point of view, personalized services for the Mobile Web typically require expensive tasks for the generation and adaptation of contents to client devices and user preferences [6, 7, 10]. For this reason, much interest of the research community has been focused on high performance systems consisting of multiple nodes to provide the user with efficient services. Besides computational cost, we should consider that most personalized services are based on information concerning the user [10] (the so-called *user profile*). The user profile may contain information about the user, such as his/her preferences, information on user click history, and lists of previous user interactions with the system. Furthermore, the user profile may contain information about the *user context*, such as user location and current activity.

Managing sensitive user information requires an infrastructure with a high security level, resulting in high maintenance costs, especially in the case of systems consisting of geographically distributed nodes. The trade-off of the solutions should be clear. The reduction of costs related to privacy management suggests to centralize services and user information on very few locations. On the other hand, performance goals suggest to spread services and information among geographically distributed nodes, with a consequent replication of sensitive data and an increase of the number of locations that must adhere to high security standards, such as the Payment Card Industry Data Security Standard (PCIDSS) [20].

Different approaches to solve this trade-off lead to a plethora of solutions for the deployment of Web systems supporting personalized services for the Mobile Web, ranging from fully centralized to peer-to-peer infrastructures. In this paper we propose an intermediate infrastructure that exploits a central component, typically a cluster, that we call *core node*. To improve the performance of the offered services, the central component is integrated with distributed *edge nodes* that are located close to the clients (Figure 1). Similar infrastructures have been proposed in distributed Web systems for traditional content generation and delivery [22, 21], but they represent a novelty in the Mobile Web scenario. In the proposed infrastructure we guarantee that the core node has the highest security standards, while we assume that it is more difficult or expensive to guarantee

Third IEEE International Conference on
Wireless and Mobile Computing, Networking and Communications (WiMob 2007)
0-7695-2889-9/07 $25.00 © 2007 IEEE

the same level of security to every edge node, that may be hosted or housed and not directly controlled. We also propose request dispatching algorithms that aim to solve the trade-off between performance and privacy. To the best of our knowledge this is the first paper that proposes infrastructures and algorithms that take into account both performance and privacy requirements, because performance and privacy issues have been typically addressed separately in literature (for example, see [24, 19, 16] for performance and [17, 12] for privacy).

Through a prototype, we demonstrate that the performance- and privacy-aware infrastructure is suitable to deploy efficient and secure personalized services for the Mobile Web. Unlike the existing solutions that separately consider performance and privacy requirements, the proposed scheme guarantees the assignment of up to the 97% of the requests accessing sensitive user information to the most secure nodes with limited penalty consequences on the response time. On the other hand, performance-oriented solutions do not guarantee the highest security for an amount of requests that is 4-5 times higher if compared to our proposal. Furthermore, privacy-oriented solutions suffer from significant performance penalty, with response times almost doubled with respect to our proposal.

The remainder of the paper is organized as follows. Section 2 describes the geographically distributed infrastructure considered in this paper. Section 3 presents the requests dispatching algorithms. Section 4 describes the prototype and the experimental testbed for the evaluation of the considered schemes. Section 5 compares experimental results by distinguishing performance and privacy. Section 6 discusses related work and Section 7 concludes the paper with some final remarks.

2 Infrastructure overview

In this section we describe the system-level details of the proposed infrastructure supporting personalized services for the Mobile Web.

This infrastructure consists of a centralized core node integrated with geographically replicated edge nodes, as shown in Figure 1. The infrastructure follows the recent trend of modern systems that exploits servers on the network edge to replicate Web-based services (possibly including personalized services), as can be observed in recent literature [22, 19, 18] and in CDN solutions [14, 12]. The use of replicated edge nodes is a viable solution also because most new mobile devices require some intermediary to access the Mobile Web.

The infrastructure model, that is described in Figure 1, is detailed in Figure 2. The core node provides a three-tier Web system with a first tier of *HTTP servers*, a second tier of *application servers* performing content generation and

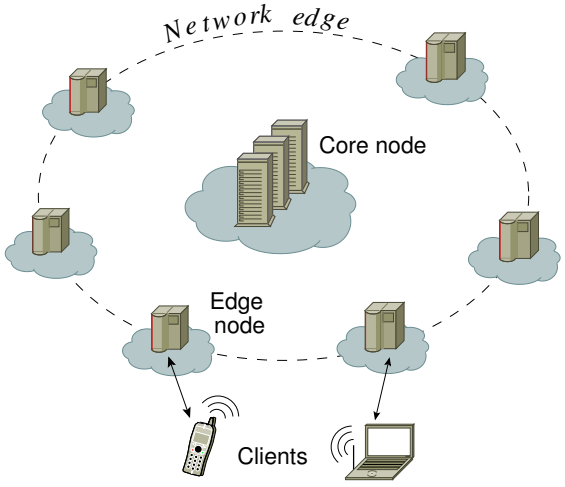


Figure 1. System supporting personalized services for the Mobile Web

adaptation, and a third tier of *back-end servers* hosting the application data. The core node also maintains a *static resources* repository and the *user profile database*. The user profiles are maintained on the core node because it guarantees high security standards while the edge nodes may be hosted or housed in locations with lower levels of physical and logical security. Each edge node consists of a two-tiered Web system with a *front-end server* and an *application server* that carries out generation and adaptation functions. As shown in Figure 2, each edge node hosts a *local data* repository with a replica of the static resources located on the core node. Every update of the static resources is propagated from the core node through push-based caching mechanisms to guarantee Web data consistency. The local data repository may also store partial user profile information that is necessary to generate and adapt Web contents on the edge nodes for a specific request. User information is cached on the edge nodes just for the user session to preserve data privacy and avoid any consistency problem related to the replication and the update of the profiles, that may change frequently in modern Web systems [13].

The trade-off between preserving data privacy on the core node and improving performance by moving services on the edge nodes is addressed through novel request dispatching algorithms. The dispatching process is performed by the edge nodes, as shown in Figure 2, and exploits a fine-grained approach at the level of *Web resource components*. Each user interaction for a Web resource generates multiple requests for the associated *components*, where each of them may range from a simple fragment of text to a multimedia resource, such as an image or an audio/video stream [18], and may require a different personalized service. The por-

Third IEEE International Conference on
Wireless and Mobile Computing, Networking and Communications (WiMob 2007)
0-7695-2889-9/07 $25.00 © 2007 IEEE

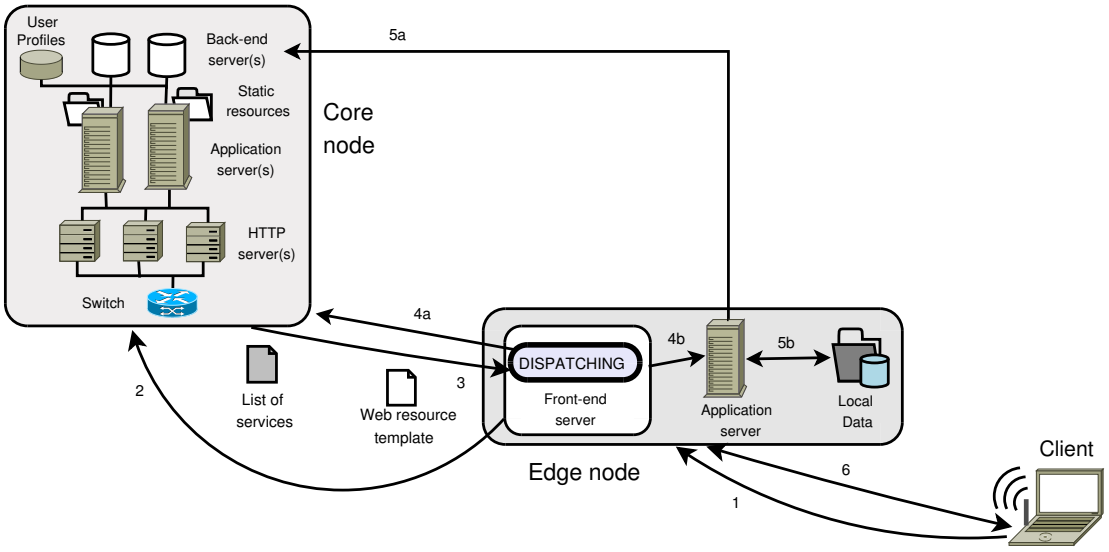


Figure 2. Detailed view of system supporting personalized services for the Mobile Web

tion of the user profile that is necessary for the personalized services determines the privacy requirements associated with that component. Throughout this paper we will consider three categories of privacy requirements. Components with *None* privacy requirements, that may be assigned to any node. Components with *Strong* privacy requirements (e.g., health information, credit card data), that must be processed on the core node since privacy is mandatory, and components with *Light* privacy requirements (e.g., user preferences, some information on user contexts), that should be assigned to the core node, but performance concerns may suggest to dispatch them to an edge node.

Figure 2 describes the steps to serve a client interaction for a Web resource. If the client request (Step 1) belongs to a new user session, the edge node contacts the core node (Step 2) to retrieve a template that enumerates the components of the Web resource and a list of the services that are required according with the user profile (Step 3). If the client request refers to an already established session, the edge node retrieves from the core node only the template of the resource. The load information about the core node is sent to the edge node along with the Web resource template. The edge node executes one of the dispatching algorithms described in Section 3 to assign the requests for Web resource components to the core node (Step 4a) or to the local edge node (Step 4b). For requests assigned to the edge node, the local application server retrieves the corresponding profile information directly from the back-end servers of the core node (Step 5a). In the same way, the application server obtains database information possibly required for the local generation process. The results of the queries to the back-end servers of the core node are cached by the edge nodes. The application server of the edge node may also interact with the local data repository (Step 5b). After the generation of all components, the Web resource is sent to the client by the edge node (Step 6).

3 Dispatching algorithms

Request dispatching is a key task for the deployment of efficient infrastructures supporting personalized services for the Mobile Web. We should consider that dispatching algorithms must be robust since they operate in a context where external and internal system conditions are subject to continuous changes [2, 15], including server load and network delays. The dispatching algorithms must also be able to handle highly heterogeneous workloads that may change in monthly, weekly or even daily patterns depending on user behavior and novel offered services. Let us analyze the information about the system and the client requests that a dispatching algorithm may access to distribute requests for Web resource components among the nodes.

For privacy concerns, dispatching algorithms may consider the security level of the distributed nodes and the privacy requirements of the Web resource components. We recall from Section 2 the three levels of privacy requirements: *Strong*, *Light* and *None*. Components with *Strong* privacy requirements must be dispatched to the core node; components with *Light* privacy requirements are preferably assigned to the core node; components with *None* privacy requirements may be dispatched to any node.

Performance is the other main issue that a dispatching algorithm should address. To this purpose, the algorithms may take into account performance-related informa-

0-7695-2889-9/07 $25.00

tion about the distributed Web system. As we have verified that most personalized services are CPU-bound operations, we consider the CPU utilization of the application servers providing personalization as the most important performance-related index of the system status [1]. However, the main results of this paper are still valid with appropriate load index changes, if we consider a system where the generation of personalized contents is based for example on disk-bound operations.

In this section we present three dispatching algorithms, namely *Performance and Privacy*, *Performance-oriented* and *Privacy-oriented*. The first algorithm represents an innovative request dispatching that aims to combine performance- and privacy-aware objectives, while the latter two alternatives separately address performance and privacy requirements.

3.1 Performance and Privacy

The *Performance and Privacy* algorithm aims to dispatch requests for Web resource components according to privacy requirements without overloading the nodes. To this purpose, the algorithm dispatches components with *Light* privacy requirements to the edge node and components with *None* privacy requirements to the core node until excessive load conditions are detected on the nodes. The CPU utilization of the edge and core nodes is the system status information that the algorithm uses to detect excessive load condition that would degrade performance. Depending on the values of CPU utilization on the edge and core nodes, we can identify four different behaviors.

1. When the edge and the core nodes have a CPU utilization U below a given threshold U_T, the algorithm dispatches the components according to their privacy requirements.
2. When the edge node utilization U exceeds the threshold U_T, but the core node load is below the threshold, the algorithm dispatches part of components with *None* privacy requirements to the core node to alleviate the load on the edge node.
3. When the core node utilization U exceeds the threshold U_T, but the edge node load is below the threshold, the algorithm follows an opposite behavior, that is dispatching components with *Light* privacy requirements to the edge node to alleviate the load on the core node.
4. When both edge and core nodes have a CPU utilization U beyond the threshold U_T, the algorithm dispatches components according with their privacy requirements (as in the case 1).

We now detail the load sharing actions for the case where the load on the edge node affects request dispatching (case 2). When the load on the core node affects request dispatching (case 3) is handled in a similar way.

When the CPU utilization U on the edge node is beyond the threshold U_T, the amount of components N' that should be forwarded to the core node is computed on the basis of the fraction $\frac{U-U_T}{1-U_T}$, that indicates how much the edge node utilization is beyond the threshold:

$$N' \Leftarrow \lfloor N \frac{U - U_T}{1 - U_T} \rfloor, \quad (1)$$

where N represents the amount of components with *None* privacy requirements. By computing N' as in Equation 1, the number of components redirected to the core node increases as the CPU utilization U of the edge node grows. For example, let us assume a threshold $U_T = 0.7$: for $U = 0.75$, that is slightly higher than the threshold, the algorithm dispatches the 16% of the components to the core node, while for $U = 0.95$, the 83% of the components is assigned to the core node. The goal is to offload the edge node that is close to saturation.

3.2 Performance-oriented

The *Performance-oriented* algorithm aims to optimize performance by maximizing the number of components processed on the edge nodes without overloading them. This algorithm exploits some of the best practices in efficient content delivery, that is, it moves computation close to the client to reduce network related delays [22, 12]. To this purpose, the Performance-oriented algorithm follows a threshold-based approach, commonly used in literature [3], to assign Web resource components to the local edge node until it reaches a certain utilization threshold U_T. When the edge node utilization is higher than the threshold, requests for some Web resource components are forwarded to the core node. The amount of components assigned to the core node is computed as in Equation 1.

3.3 Privacy-oriented

The *Privacy-oriented* algorithm aims to satisfy privacy requirements for the totality of the Web resource components. This algorithm assigns every component with *Light* and *Strong* privacy requirements to the core node, while the components with *None* requirements are typically served by the edge node. It is worth to note that in scenarios where the majority of the components have *None* privacy requirements, assigning all them to the edge node could easily overload it. Indeed, we assume that the edge nodes of the proposed infrastructure have limited computational capacity with respect to the core node. Hence, to avoid excessive load conditions on the edge node in our system we choose

to assign to that node at most the 50% of the resource components and to dispatch the remaining requests with *None* privacy requirements to the core node.

4 Experimental setup

We implement a prototype providing personalized services for the Mobile Web that resemble the services offered by a personalized portal Web site for fixed and mobile clients.

The prototype generates and adapts Web contents on the basis of the user device, preferences, context and location through three main services. 1) *Aggregation of RSS feeds:* service that dynamically aggregates and converts RSS-XML code to HTML depending on the user preferences. 2) *Context-sensitive banner insertion:* service that selects banners from a database according to user interests, location and current activity and inserts them into the Web resource. 3) *Adaptation to user device and context:* service that tailors HTML code and embedded images of the Web resource according to the user context. For example, the case of a driving user may require services, such as text-to-speech conversion [4], to access Web contents without the need to read the information.

The above services are ordered by increasing computational requirements, that may involve service times of different orders of magnitude, ranging from few milliseconds for a banner insertion up to hundreds of milliseconds for the adaptation of an embedded image [7]. We define a workload model where the requests are evenly distributed among the three offered services. We use synthetically generated traces, since none of the existing Web benchmarks includes features for personalized services based on user profile and context information. The requests are divided into sessions that are initiated at the rate of 5 sessions per second. Each session is related to a different user and contains requests for 5 Web resources on average, where each resource typically consists of 10 components.

In our experiments we focus on scenarios characterized by different mixes in the privacy requirements of client requests. In every scenario we consider that 10% of requests is characterized by *Strong* privacy requirements. We then define three scenarios with an increasing amount of resource components with *Light* privacy requirements where percentages go from 10% to 40%, to 70%.

For the experiments we consider a distributed system consisting of a core node and four edge nodes. The core node is a three-tier Web system with two Web server nodes as the front-end, four application servers and two back-end servers. Each edge node is composed by two servers: a front-end Web server node and an application server. Since the offered services are characterized by a significant computational cost, most system load weights on the application servers. For a fair comparison among the dispatching algorithms with no influence due to the specific architectural choices, we use the same number of application servers on the core and the edge nodes that in such way provide the same computational capacity.

We emulate wide area network effects between the edge and the core nodes through the *netem* packet scheduler (part of the Linux kernel) that creates a virtual link between the edge and the core nodes. We consider three network scenarios where the mean value for the delay on the edge-to-core links is set to 10, 40, 100 ms. The emulated WAN effects include also packet loss (set to 1%) and bandwidth limitation (10Mbit/s) for a full WAN emulation [26]. Unless otherwise indicated, throughout the paper the experiments are referred to the scenario with mean edge-to-core delay equal to 10 ms.

5 Experimental results

We evaluate the proposed infrastructure and dispatching algorithms on the basis of two main indexes: performance and privacy.

The index to evaluate performance results is the resource response time on the client nodes, that is measured as the time between the client request and the arrival of the whole Web resource. The privacy index considers how frequently the dispatching algorithms assign components with some privacy requirements to the edge nodes, that are considered less secure than the core node. To this purpose, we evaluate the *dispatching mismatch*, that is defined as the amount of Web resource components with *Light* privacy requirements that are assigned to an edge node. It is worth to note that components with *Strong* privacy requirements do not contribute to the dispatching mismatch because all considered algorithms assign them to the core node. From the privacy point of view, we consider best the scheme providing the lowest percentage of dispatching mismatch.

5.1 Infrastructure evaluation

We start our analysis by evaluating whether the proposed infrastructure may effectively support personalized services for the Mobile Web. We take into account the impact on performance of different network delays on the links between the edge and the core nodes. Indeed, whenever a Web resource component is processed on the core node, it has to pay an additional delay with respect to a component that is processed on the edge node. Hence, it is interesting to evaluate the infrastructure performance as the delay on the edge-to-core links grows. For this analysis we refer to the privacy scenario where the amount of components with *Light* privacy is equal to 40%. Experiments carried out with

Third IEEE International Conference on
Wireless and Mobile Computing, Networking and Communications (WiMob 2007)
0-7695-2889-9/07 $25.00 © 2007 IEEE

other privacy scenarios do not change the main conclusions of this analysis.

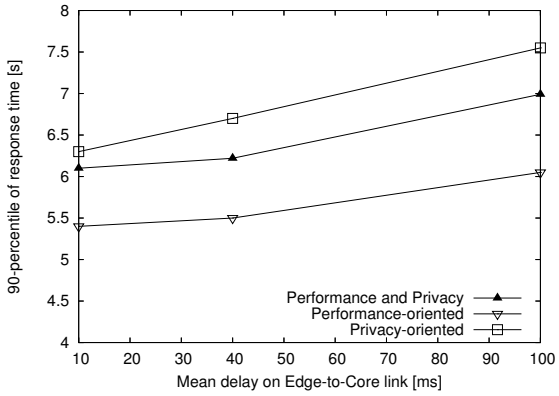


Figure 3. Effect of network delays

Figure 3 presents the 90-percentile of the response time achieved by the proposed infrastructure as a function of the mean network delay on the edge-to-core links. The three curves refer to the dispatching algorithms described in Section 3.

As a first result we have a confirmation of the effectiveness of the proposed infrastructure. Even if the provided services are computationally expensive, the response time remains in the order of 5-7 seconds for an edge-to-core delay below 40 ms. Furthermore, even in the case of high delay (100 ms), the response time stays below 8 seconds, which may be considered acceptable for the users [9]. A further important result is the confirmation of the non negligible impact of network delay on performance, regardless of the dispatching algorithm. As the mean delay on the edge-to-core link passes from 10 ms to 100 ms, the performance degradation on the 90-percentile of the response time is in the order of 20% for every considered algorithm, as testified by the almost parallel curves of Figure 3. The curves allow a first performance comparison of the algorithms implemented on the proposed infrastructure. For every network delay, the Performance-oriented algorithm has the lowest response time, while the Privacy-oriented algorithm achieves the highest response times. The proposed Performance and Privacy algorithm gets intermediate performance between the other two alternatives. We analyze the motivation of this result in the following section.

5.2 Performance of the dispatching algorithms

Figures 4(a), 4(b), and 4(c) show the cumulative response time of the three dispatching algorithms for scenarios where the amount of components with *Light* privacy is equal to 10%, 40% and 70%, respectively. For every considered workload, we confirm the observation of Section 5.1: the Performance- and Privacy-oriented algorithms are the best and the worst performing solutions, respectively; the proposed Performance and Privacy algorithm achieves intermediate results.

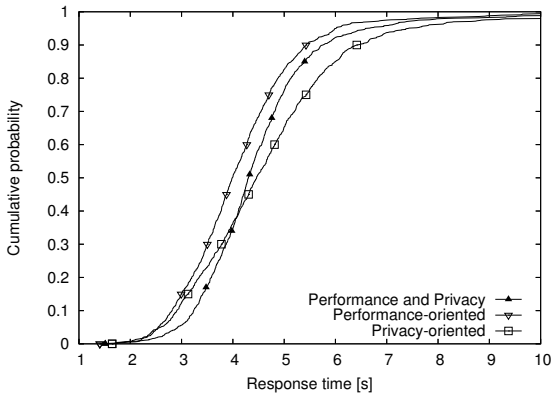
The Performance-oriented algorithm achieves good performance since it exploits as much as possible the available computational power on the edge nodes. Performance results are consistent with every privacy scenario because this algorithm discards any privacy-related information.

The Privacy-oriented algorithm achieves similar performance for the scenarios in Figures 4(a) and 4(b), since the amount of components with *Light* privacy requirements (10% and 40%, respectively) allows the algorithm to achieve a fair load sharing between edge and core nodes. However, when most resource components have *Light* privacy requirements (PL-components=70%) the Privacy-oriented algorithm achieves really poor performance, as shown in Figure 4(c)). The motivation is twofold. First, the uneven load sharing places a significant amount of resource components on the core node, thus augmenting the delays on this node. Second, the components processed on the core node add a non-negligible latency in the response time due to the edge-to-core network delays.

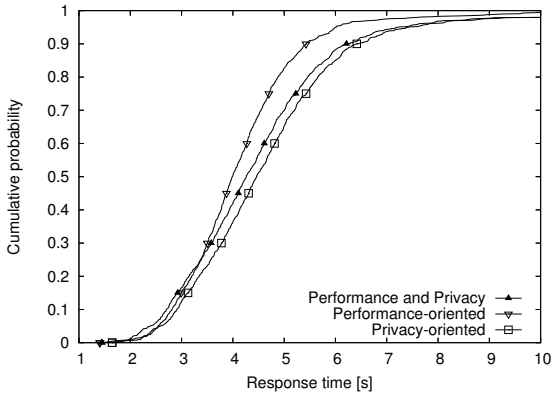
If we pass to analyze the response times of the Performance and Privacy algorithm, we observe that they are similar to those of the Performance-oriented algorithm for a low percentage of components with *Light* privacy (Figure 4(a)). Then, the response times grow for increasing amounts of components with *Light* privacy. The reason for this behavior is that the Performance and Privacy algorithm tends to place an increasing amount of computation on the core node as the components with *Light* privacy augment, with a consequent increase of the network delay contribution to the response time. However, it is important to note that the proposed algorithm guarantees much better performance than the Privacy-oriented alternative when a significant amount of components have *Light* privacy requirements: the performance gain on the 90-percentile is over 24% of the response time for when these components are equal to 70%.

5.3 Privacy of the dispatching algorithms

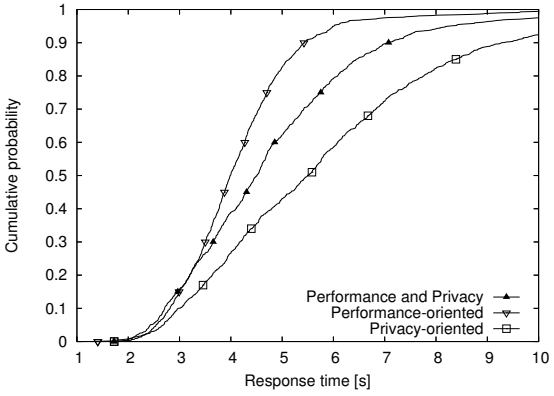
Our analysis has been focused so far on the performance of the proposed scheme. We now consider privacy-related results to evaluate to which extent the dispatching algorithms may preserve user data privacy. Table 1 shows the percentage of dispatching mismatch for all the considered algorithms and the three privacy scenarios. The Privacy-oriented algorithm, which always returns the optimal privacy-aware dispatching, obtains a 0% mismatch for every scenario. On the other hand, the Performance-



(a) *Light* privacy 10%



(b) *Light* privacy 40%



(c) *Light* privacy 70%

Figure 4. Cumulative distribution of response time of the dispatching algorithms for different privacy scenarios

oriented algorithm, that applies a privacy-blind dispatching, causes a not acceptable percentage of mismatches, which is up to 43.8% for an amount of components with *Light* privacy equal to 70%.

Table 1. Dispatching mismatch [%]

Algorithm	*Light* privacy		
	10%	**40%**	**70%**
Performance and Privacy	2.5 %	4.6%	14.7%
Performance-oriented	11.5%	25.2%	43.8%
Privacy-oriented	0 %	0%	0%

If we analyze the results of the Performance and Privacy algorithm, we observe that the achieved dispatching mismatch increases as the percentage of components with *Light* privacy grows. This result can be explained by considering that the trade-off between performance and privacy forces this algorithm to accept dispatching solutions that are suboptimal from the privacy point of view to preserve an adequate level of performance. However, the dispatching mismatch of the Performance and Privacy algorithm is significantly lower with respect to the Performance-oriented algorithm that does not consider any privacy-related information (reduced to one third for *Light* privacy equal to 70%). This represents an important result because it is achieved by avoiding the severe penalization on the response time caused by the Privacy-oriented behavior for significant amounts of components with *Light* privacy, as discussed in Section 5.2.

6 Related work

The Mobile Web has been recognized as a fundamental challenge by multiple authors [25, 23], but the importance of preserving privacy of user information while providing personalization services has been pointed out only by recent literature [8, 17]. Multiple distributed and parallel systems and related dispatching algorithms have been proposed with the main goal of improving performance, while scarce or no attention has been devoted to privacy requirements in request dispatching decisions. In many cases, privacy is not considered at all, while some systems adopt solutions with straightforward dispatching that strongly limit the possibility of distributing personalization tasks.

When the infrastructure is based on a cluster Web system [5], privacy requirements are not taken into account because this architecture may guarantee high levels of security for any of its nodes.

Other infrastructures for the delivery of Web content that are more similar to our proposal adopt a geographical distribution of nodes, with multi-clusters or single-cluster integrated with geographic replicated servers or even CDNs [21, 19]. These systems rely on request dispatching algorithms that are based on factors like network and geographic proximity, network link status or server load. We

Third IEEE International Conference on
Wireless and Mobile Computing, Networking and Communications (WiMob 2007)
0-7695-2889-9/07 $25.00 © 2007 IEEE

introduce in this field a novel concept of algorithm for request dispatching that addresses both performance and privacy issues.

The need of taking into account privacy in the design of scalable distributed architectures is confirmed by P3P (Platform for Privacy Preferences) [11]. It is a W3C proposal that suggests a mechanism for Web sites to encode their privacy policies in a standardized format that can be easily retrieved and interpreted by user agents. However, these studies are more tailored to a server-side approach for the generation of personalized Web content rather than to the intermediary-based model for Web content adaptation considered in this paper. Some recent studies, that replicate the application logic on the edge servers [12, 22], propose the specialization of a subset of servers to handle a specific set of services. This mechanism may be used to preserve data privacy, but it has the drawback of limiting the flexibility of the infrastructure because only some nodes may provide personalization services. On the other hand, we propose a more flexible infrastructure demonstrating that privacy and performance requirements may be pursued together.

7 Conclusions

In this paper, we propose a distributed infrastructure to support personalized services for the Mobile Web and a related set of request dispatching algorithms. The infrastructure, that is composed by a central core node and geographically distributed edge nodes, exploits an innovative dispatching algorithm that takes into account both performance and privacy issues in the service of client requests.

Our experiments demonstrate that our proposal can successfully combine performance and privacy requirements in providing personalized services for the Mobile Web. For every considered workload and network scenario, the proposed scheme preserves from 85% to 97% of the requests privacy requirements, with a limited penalization on the response time if compared to solutions that just aim to optimize the user-perceived performance.

References

[1] T. Abdelzaher, K. Shin, and N. Bhatti. Performance guarantees for Web server end-systems: A control-theoretical approach. *IEEE Trans. on Parallel and Distributed Systems*, 13:80–96, Jan. 2002.

[2] V. A. Almeida and D. A. Menasce. Capacity planning: An essential tool for managing Web services. *IT Professional*, 4:33–38, July 2002.

[3] M. Aron, P. Druschel, and W. Zwaenepoel. Efficient support for P-HTTP in cluster-based Web servers. In *Proc. of USENIX 1999*, Monterey, CA, Jun. 1999.

[4] M. Barra, R. Grieco, D. Malandrino, A. Negro, and V. Scarano. Texttospeech: A heavy-weight edge service. In *Proc. of 12th WWW Conference*, Budapest, HU, 2003.

[5] V. Cardellini, E. Casalicchio, M. Colajanni, and P. S. Yu. The state of the art in locally distributed Web-server systems. *ACM Comput. Surv.*, 34(2):263–311, 2002.

[6] E. Cecchet, A. Chanda, S. Elnikety, J. Marguerite, and W. Zwaenepoel. Performance comparison of middleware architectures for generating dynamic Web content. In *Proc. of 4th Middleware Conference*, Jun 2003.

[7] S. Chandra. Content adaptation and transcoding. *Practical Handbook of Internet Computing*, 2004. (Munindar P. Singh ed.), Chapman Hall & CRC Press.

[8] R. K. Chellappa and R. G. Sin. Personalization versus privacy: An empirical examination of the online consumer's dilemma. *Information Technology and Management*, 6(2-3), April 2005.

[9] W. Chiu. Design pages for performance. *IBM HVWS*, 2001.

[10] M. Colajanni, R. Lancellotti, and P. Yu. Distributed architectures for web content adaptation and delivery. *Web content delivery*, 2005. (Tang, Xu, Chanson eds.), Springer.

[11] L. Cranor. *Web Privacy with P3P*. O'Reilly, 2002.

[12] A. Davis, J. Parikh, and W. E. Weihl. EdgeComputing: extending enterprise applications to the edge of the internet. In *Proc. of WWW'04 Alternate track*, pages 180–187, 2004.

[13] M. Eiriniaki and M. Vazirgiannis. Web mining for Web personalization. *ACM Trans. on Internet Technology*, 3(1), 2003.

[14] Edge Side Includes, 2002. `http://www.esi.org/`.

[15] S. Floyd and V. Paxson. Difficulties in simulating the Internet. *IEEE/ACM Trans. on Networking*, 9(4):392–403, 2001.

[16] L. Gao, M. Dahlin, A. Nayate, J. Zheng, and A. Iyengar. Application specific data replication for edge services. In *Proc. of 12th WWW Conference*, Budapest, HU, 2003.

[17] R. Hull, B. Kumar, D. Lieuwen, P. F. Patel-Schneider, A. Sahuguet, S. Varadarajan, and A. Vyas. Enabling context-aware and privacy-conscious user data sharing. In *Proc. of MDM'04*, 2004.

[18] A. Iyengar, L. Ramaswamy, and B. Schroeder. Techniques for efficiently serving and caching dynamic Web content. *Web content delivery*, 2005. (Tang, Xu, Chanson eds.), Springer.

[19] M. Karlsson. Replica placement and request routing. *Web content delivery*, 2005. (Tang, Xu, Chanson eds.), Springer.

[20] Master Card Int'l. *Payment Card Industry Data Security Standard*, Jan. 2005.

[21] M. Rabinovich and O. Spatscheck. *Web Caching and Replication*. Addison Wesley, 2002.

[22] M. Rabinovich, Z. Xiao, and A. Aggarwal. Computing on the edge: A platform for replicating Internet applications. In *Proc. of WCW'03*, Hawthorne, NY, Sept. 2003.

[23] D. Saha and A. Mukherjee. Pervasive computing: A paradigm for the 21st century. *IEEE Computer*, 36(3), Mar. 2003.

[24] S. Sivasubramanian, M. Szymaniak, G. Pierre, and M. van Steen. Replication for Web hosting systems. *ACM Computing Surveys*, 36(3):291–334, 2004.

[25] G. C. Vanderheiden. Anywhere, anytime (+ anyone) access to the next-generation WWW. *Computer Networks and ISDN Systems*, 8(13), Sep. 1997.

[26] R. Zhang, C. Hu, X. Lin, and S. Fahmy. A hierarchical approach to Internet distance prediction. In *Proc. of ICDCS'06*, Washington, DC, USA, Jul. 2006.