Cloud Aided Internet Mobility

Ping Zhang, Arjan Durresi
Department of Computer and Information Science,
Indiana University Purdue University Indianapolis
Indianapolis, IN, USA

Raj Jain
Department of Computer Science and Engineering,
Washington University in St. Louis
St. Louis, MO 63130, USA

***Abstract*—The Internet is becoming increasingly mobile. Although several mobility solutions are proposed, none of them has been largely deployed. We propose a new system that can facilitate and support mobility on the Internet. The new mobility support service will be offered as a value-added service by mobility service providers to paying mobile customers. Therefore, the proposed solution is economically viable. We propose to design our mobility management system on cloud computing. Furthermore, we explore the architectural tradeoffs among QoS, economic viability, security and privacy of various cloud aided designs. Our simulation results show how various system architectures could be used to satisfy different requirements.**

I. Introduction

Mobility is a major requirement for the future Internet. Recent studies predict that mobile data traffic will double every year through 2014, increasing 39 times between 2009 and 2014 [1] with cell phones becoming the major component of the new communication and computing platform [2].

Mobility support for the Internet is a well-known issue and a number of solutions have been proposed [3], [4], [5], [6], [7], [8], [9]. However, until now none of the proposed Internet mobility solutions have been largely deployed, and the only available method to access Internet in mobile manner is through cellular data networks.

We believe that several critical economic flaws have also made many "technically feasible" mobility solutions infeasible in real world. *First*, existing solutions, such as Mobile IP [8], require modifications on access networks. But, such ubiquitous deployment of network changes do not offer sufficient economic incentives, especially for service providers. *Second*, existing solutions require that all Internet users pay the cost of deployment and operations of the given mobility support, even though a large portion of users might not be mobile. *Finally*, in existing solutions, while technical collaborations among involved service providers are required, there is no mechanism to split the revenues among them.

We have proposed Mobility Support Service (MSS) protocol to manage Internet mobility [10]. MSS will be offered by service providers dedicated to mobility, called Mobility Service Provider (MSP). MSP role could be played by existing service providers too. MSS will be offered as a value-added service to users who are willing to pay for it. Therefore, mobility support will generate its own revenue and justify the business and investments of MSPs. MSS does not require any change on access networks, existing network infrastructure, legacy applications, and operating systems.

Cloud computing services already are being used to enhance various Internet functionalities. For example, Amazon Route 53, built on Amazon Cloud Computing infrastructure, implements a DNS service. In this paper we extend the design of MSS by using cloud computing platforms. Furthermore, we show that cloud based architectures offer interesting tradeoffs among performance, security, privacy and economic viability. Cloud computing not only can improve the technical performance of our MSS, but most importantly, can make economically attractive the business of MSS.

The paper is organized as follows: In Section II we review related works. Section III describes our MSS. In Section IV, we discuss location management and its economics. In Section V, we discuss the cloud aided MSS, including various architectures, their advantages in the tradeoffs between performance, security and privacy, as well as replication algorithm for cloud based MSS. In Section VI, we explore the tradeoffs between performance and privacy offered by the proposed architectures and algorithms. We conclude in Section VII.

II. Related work

Mobile IP (MIP) [11] and its enhancements [12] are among the most popular solutions proposed to support mobility. MIP-like solutions require modifications on access networks and collaborative support from both home and foreign service providers [13], [9]. Therefore, both agents should have contracts that governs their collaboration, including sharing of revenues from their services, but such contracts are very difficult to implement among random pairs of agents (service providers.)

In Host Identity Protocol (HIP) [6], [14], Host Identity (HI) are used. HI is initially acquired by DNS lookup [15], and mobile node keeps updating peers and DNS record during move. But DNS cannot support mobility, because its updates are slow [16]. For highly mobile nodes a Rendezvous Server (RVS) is proposed [17], but its distributed implementation is not discussed. Furthermore, HIP is proposed to be an Internet service, similar to DNS, therefore paid by all Internet users, mobile or not. In addition, MSS could be used as a framework to implement HIP.

The Locator/Identifier Separation Protocol (LISP)[7], [18] uses Endpoint Identifiers (EIDs), and addresses, called Routing Locators (RLOCs). EID-to-RLOC mapping are performed at the RLOC router and routing is accomplished by tunneling

978-1-4673-3122-7/13/$31.00 ©2013 IEEE

between RLOC routers. Several overlay mobility solutions have been proposed [19], [5], [20], [21], [22].

III. Mobility Support Service (MSS)

Three entities interact in MSS protocol [10]: mobile subscribers, Mobility Service Providers (MSPs), and communicating peers. Subscribers or customers are mobile users who are provided with unique IDs and pay their MSP for the corresponding mobility services. An MSP is an organization that manages IDs, mobility and related functions for its subscribers. Communicating peers or simply users are Internet endpoints who use the MSP services to lookup for the current subscriber's address and related information. The level of MSS distribution will depend on the desirable tradeoffs among QoS, security and associated costs.

In MSS protocol, IP addresses are used as locators representing the current point of attachment. Fig. 1 illustrates the architecture of MSS. An MSS client application, called Mobility Support Layer (MSL), is installed on the nodes of mobile subscribers and users that would use MSS. Applications that want to utilize MSS are registered manually or automatically at the local MSL. When a registered application initiates a connection to a subscriber, MSL intercepts the connection setup system call and resolves (lookup) the destination ID to the corresponding current IP address by retrieving it from the MSS. Therefore, the ID resolution is transparent to applications. Furthermore, MSL can obtain further information from MSS, such as security keys to be used in communications with the corresponding mobile subscriber. Then initiating MSL will negotiate with the MSL on the mobile node, and will create the appropriate connection to current subscriber's IP address. Similar tasks are performed at the subscriber's side too. When both sides are mobile nodes, the ID resolution would be performed in reverse direction as well, and the two corresponding MSPs may not be necessarily the same. No special routing requirement is needed to deliver the packet to the destination IP address, and the routing mechanism is completely unaware of the mobility support.

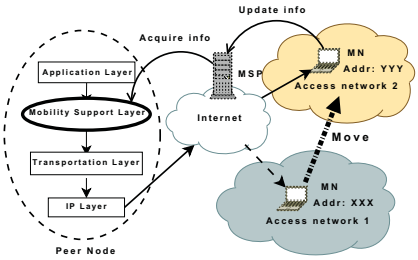

Fig. 1. The architecture of MSS

IV. Replicated Location Management

Location management, as a crucial building block of Internet mobility support, is evaluated by two major criteria: availability and latency to access the profiles of subscribers. The key to make location management reliable and efficient is to distribute profiles properly, that means as close as possible to the place where they are used.

We use the terms: "lookup" for profile query and "update" for profile modification. For a given lookup request, it would be desirable to get the reply from a replica near by rather than from one far away. Furthermore, when we have multiple replicas of a given profile, we need to update all of them when a modification is made, which is called "synchronization." Each synchronization will induce certain amount of overhead.

The proposed cloud based MSS enhances the performance and availability of location management system by replicating profiles of a mobile subscribers at multiple locations, and dynamically adjusting the replica distribution to balance the cost of implementing replication and overall performance. The primary benefit of replication is that MSS subscribers and their peer communicators could have a guaranteed service performance, such as latency bound for address update and lookup. However, the replication has to be handled carefully as a tradeoff among the improvements in user's QoS (lookup), subscriber updates, and replica synchronization and replica distribution costs.

Our approach in MSS replication is based on previous work in distributed systems, databases and cellular mobility systems [23], [24]. To be able to optimize the replication process, we developed a cost function that captures various network costs and user's QoS.

Economic Model and Cost Functions

We developed a cost function to be used by replication algorithms in exploring tradeoffs among performance, QoS and various costs: communication, computing, security, reliability and privacy costs both on cloud and on hybrid architectures. We start by introducing a simple cost function that reflects our goal of incorporating the effect of replications in improving user's QoS as well as the cost in replica synchronization and replica implementation by distributed servers. Using this model, we then outline its possible use for exploring strategies to realize different objectives that balance user goals and network costs.

Cloud computing with its flexible business model is the ideal solution to cover the initial costs of MSS. So the MSP could invest depending on the number of clients. The operation costs of MSS include financial and technical costs. Financial cost includes resources to acquire computation and communication capacities, such as depreciation of equipment, management and maintenance, rent of network links, usage of cloud services, etc. Technical costs include factors that reduce the QoS perceived by users, such as the latency caused by signaling messages and server processing on MSS.

Each node of the topology graph represents a subnet hosting a MSS server, and the requests from subnetworks without MSS servers are combined into the nearest ancestor network that has a server. Then for a signaling message traveling through two (logically) adjacent subnets: from x to y, we define the cost $c(x, y)$ as:

$$c(x,y) = \sum_{\forall}(\kappa_1 metric_1(x,y) + \kappa_2 metric_2(x,y) + ...) \quad (1)$$

The value of each metric needs to be normalized and converted to a neutral unit, and then summed together, where $metric_1(x,y)$ is the neutral unit conversion of cost based on *metric* or *policy 1*, and κ_1 is the weight coefficient of $metric_1(x,y)$. For example, $metric_1$ could stand for latency from subnet x to y, and $metric_1$ could be fees associated with usage of link from x to y. *Metrics* will include communication delay, processing delay, monetary costs of processing and communications both in public Internet and cloud.

A simple model for the cost of a lookup request for a mobile node A, l^A, consists of three parts: cost $c(x,y)$ for the request message traveling from requesting node in subnet x to corresponding server in subnet y; cost $c(y,x)$ for response message traveling back in the reverse direction; and possible propagation cost $Prop_l^A(y)$ of notifying necessary servers and possibly adjusting replica states, if any, originating from subnet y. Therefore:

$$l^A(x) = c(x,y) + c(y,x) + Prop_l^A(y) \quad (2)$$

Similarly, the cost u^A of an update request sent by node A can also be divided into three parts: cost $c(m,n)$ for the request message traveling from A in subnet m to corresponding server in subnet n; cost $Sync^A(n)$ of all sync messages traveling to synchronize profiles where the first one originates from subnet n, and cost of propagation $Prop_u^A(n)$ which is similar to $Prop_l^A(y)$. Therefore:

$$u^A(m) = c(m,n) + Sync^A(n) + Prop_u^A(n) \quad (3)$$

We define L^A and U^A are sum of serving all lookup and update requests for mobile node A during a given period of observation, such as:

$$L^A = \sum_{\forall i} l^A(x_i) = \sum_{\forall i} \left(2 \times c(x_i, y_i) + Prop_l^A(y_i)\right) \quad (4)$$

and

$$U^A = \sum_{\forall j} u^A(m_j) = \sum_{\forall j} \left(c(m_j, n_j) + Sync^A(n_j) + Prop_u^A(n_j)\right) \quad (5)$$

Therefore $L^A + U^A$ is the cost function or optimization goal of Location Management for node A. In addition, we need also consider infrastructure cost. When employing public cloud the cost computation changes further; usually cloud users are charged only based on the amount of usage, such as the number of VM, CPU time, storage, in/out traffic, etc. So, the basic cost includes:

$$C_{Basic} = \kappa_L L^A + \kappa_U U^A + \kappa_I Infra + \kappa_C CloudUsage \quad (6)$$

Furthermore, we consider how much redundancy the system needs for both fault tolerance and resilience against security attacks, which includes the cost for other security related investments such as firewalls, IDS, etc. The redundancy and security cost has a similar structure as the basic cost. Therefore, the total cost is:

$$C_{Total} = \kappa_B C_{Basic} + \kappa_{RS} RedundancySecurity \quad (7)$$

Finally, the profit of MSP from the service will be the sum of the gains from each customer, which is the difference between the revenue from all customers subtracting the total cost and potential penalties for not satisfying the signed SLA with customers [25].

$$G_t = \sum_{\forall i} G(S_i) = \sum_{\forall i} Rev_i - \sum_{\forall i} Pen_i \quad (8)$$

There are many possible tradeoffs that regulate the interplay between QoS, trustworthiness and economics in equation 8. For example, if the QoS, such as look up or update delay, or the offered security and privacy could go down because the MSP is not investing in enough resources, that might lead to higher penalties for violating the signed SLA. Therefore, a MSP can use this model to make the most appropriate choices to increase the longtime profit and customer satisfaction.

We have developed two MSS replication algorithms for on-premise architecture that maximize C_{Total} [10]. In this paper we develop a new replication algorithm to be used in cloud and hybrid scenarios. These problems are NP-hard problems so optimization is usually conducted on reduced problems or targeted at sub-optimal solutions.

V. Cloud based Architectures for MSS

Cloud computing can improve both technically and economically the MSP's activities. By using the flexible and payed as used cloud services, an MSP could tune better and cheaper its QoS. For example, a larger number and better distributed MSS servers would be needed for better availability and QoS. Therefore, a good MSS could be offered only by big service providers. But by using various cloud platforms, and leveraging on their distribution, even modest service providers could offers MSS with high distribution level. As the cloud services are payed based on their use, MSS business will scale very well with the number of subscribers. In particular, cloud is very suitable for a centralized replication algorithm, which has global knowledge.

A. Privacy and Cloud

Security and privacy are the major concerns related to the use of cloud computing [26]. In particular the MSS protocol uses very sensitive data, including the current and history of user location (IP address). Therefore, the considerations about the security and privacy will be leading factors in designing and using MSS over cloud computing. Other sensitive data, part of user's profile include: contact lists, various security keys, payment information, and so on. We can encrypt the

above information before sending it to cloud. However, sensitive information could be extracted even from encrypted data. For example, someone in cloud could analyze the traffic, as as look ups, correlate such information with other type of information and finally be able to obtain user's location.

Various combinations cloud and on-premise architectures offer different levels of privacy and security protection. However, if no security and privacy vulnerability can be tolerated, the best choice is not to use cloud at all for MSS, and use the on-premise architecture.

B. MSS all in Cloud

In this architecture the cloud takes all responsibilities and tasks of MSS, which includes subscriber tracking, query serving, authentication, and access control. Therefore, there is no privacy from cloud. Furthermore, the MSS security depends totally on the cloud security. In this model, all of MSS's data and algorithms are held in cloud, and both subscribers and their peers directly connect to the service through interface provided by cloud service provider.

For reliability concerns, the cloud would actively maintain more than one copy of the data container, and these replica will be stored at different locations to satisfy cloud service provider's defined policy about reliability and security. Performance could be also a criteria of selecting replica locations.

Whereas this is the most intuitive and simple cloud based MSS, it is suitable only when the cloud provider is also MSP, and takes all the corresponding responsibilities.

C. MSS in Tree Hybrid Cloud

In this architecture, illustrated in Fig. 2, MSS is implemented partially in cloud and partially on servers controlled by MSP. Subscribers and users do not interact directly with the cloud. Therefore, information regarding the origin of updates and lookups could be hidden from the cloud. Furthermore, the data will be encrypted before being sent to cloud, which increases the defences against security and privacy attacks. Another advantage is that the MSP has more freedom to optimize the MSS service in various areas, and not depend totally on the cloud performance, both in computing and network access.

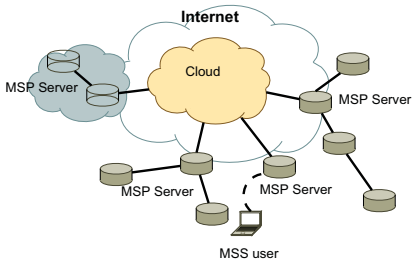

Fig. 2. Tree Hybrid Architecture of MSS

MSP decides the level of visibility of cloud. For parts of the MSS network of servers, the cloud might have full visibility, therefore run the replication algorithm considering all servers of that zone. But autonomous zones are allowed in the tree model. In such zones the MSS server may hide its structure of the network of servers from cloud, such as as the two servers in shade in Fig 2. These sub servers are completely controlled by their parent server, and all requests originated from these sub servers will be marked as from the parent server. Cloud then would treat all the subtree as a single server, and the parent server would be responsible of managing replica distribution in its realm.

The major benefit of the tree hybrid architecture is that it protects against traffic analysis done by cloud. For example, while a low level MSS server could cover a small geographical area, such as a town, or a small range of IP address, a parent server could cover a much larger geographical of IP address range areas, such as a big city or larger area. So, when cloud has full visibility of MSS servers, it will know the details of users location (IP and geographical), but when the replica is managed in an autonomous zone, the cloud would only know that the user is in that city or in that state. Therefore, the privacy risk is inversely proportional to the size of the autonomous zone.

D. Sliced Time Replication Algorithm for Hybrid Architecture

This is a simplified offline algorithm combined with thresholds in cloud. Furthermore, the algorithm adapts to request patterns dynamically. First, request sequences are divided into blocks separated by each update request. Then, we count the number of lookup requests between every two updates, which is a score of the node $C(j,t)$. For the last update, the most recent lookup request is viewed as the end of counting. The action being taken depends on the type of the most recent request upon which the replication algorithm is called. For an update request, we should either do nothing or remove replicas that falls below threshold, since synchronization along with this request will only impair all replicas and candidates' saving score. For a lookup request, we should either do nothing or setup replicas at most strategy places, because the arrival of this request only promotes the possibility of new lookup requests originated from this server. Therefore the algorithm would only subtract the score of all nodes and remove unqualified replica upon woken on a update request, and only increase score of nodes receiving lookup requests and setup new replica when nodes score exceed threshold upon woken on a lookup request (lookup requests may be accumulated for one iteration of replication algorithm).

Usually it is difficult to find optimal values for thresholds, and a static threshold also lacks flexibility for various scenarios. We employ history information to help dynamically adjust threshold. The statistics' history is also summarized based on blocks combined with timestamp. MSP would manually give a cost exchange ratio $R_x(N_j)$ between lookup and update request for node N_j, depending on interest of MSP such as server's location, capacity, bandwidth etc. For example, $R_x(N_j) = 3$ means MSP decides that cost of serving three lookup at N_j is equal to cost of one update. This R_x is called original threshold. History analysis algorithm would

sum average number $H_x(N_j, t)$ of lookup of N_j for recent block and with similar timestamp T of past days, weeks, and months. Then balanced threshold would be $R_x - H_x(N_j, t)$. A brief algorithm is shown in following. Set_R is the set of replica nodes.

Algorithm 1 Sliced time algorithm

```
if Request is update then
  for all subtrees T_i do
    for all nodes N_j of T_i, starting from leaf do
      Move history statistic one block or corresponding
      timestamp forward
      Update H_x(N_j, t)
      Remove subscriber's previous server from Set_R
      Subscriber's current server ← Set_R
      Subtract R_x(N_j) from C(j, t)
      if C(j,t) + H_x(N_j,t) < R_x(N_j) and N_j ∈ Set_R
      then
        Remove N_j from Set_R
        Add C(j, t) to parent of N_j
      end if
    end for
  end for
else
  for all nodes N_j receive new lookup requests do
    Move history statistic one block or corresponding
    timestamp forward
    Update H_x(N_j, t)
    Increase C(j, t)
    if C(j,t)+H_x(N_j,t) > R_x(N_j) and N_j ∉ Set_R then
      N_j → Set_R
    else
      Add lookup count to parent of N_j
    end if
  end for
end if
```

The algorithm complexity is $O(N^2)$ for a single subscriber, where N is the number of MSS servers.

VI. Simulation and Analysis

We used Inet Topology Generator [27] to generate simulation network topologies. Then we randomly picked 100 ASes as candidates for deploying MSS server, and aggregated the remaining ASes into subnets as shown in Fig. 3, in which the network consists of subnets and links connecting subnets in roughly four tiers.

For cloud MSS simulation, the white server in the middle represents the place of cloud, i.e. at a tier-1 AS. For the simulation duration, each mobile node moves 20 times, only at leaf level subnets, and during simulation one mobile node will be looked up 200 times, for a 1:10 update/lookup ratio. The update and lookup pattern for each mobile node is random generated, and then 200 rounds of trace are generated based on the pattern. Each node moves randomly complying to Binomial distribution with $p = 0.5$ every time, and random

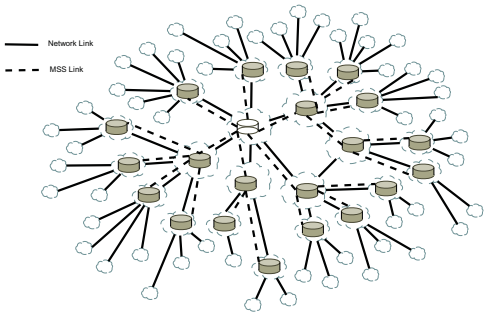

Fig. 3. Simulation network topology

lookup request locations comply to a uniform distribution across all leaf subnets. Requests in each generated round have random variance from the pattern. We use 86400 seconds (or 24 hours) as total timestamp for a round. Lookup timestamp variance is 1800 (equivalent to 41.7%) and update is 900 (equivalent to 208%). Location variance is 2.81% for update request and 1.40% for lookup requests. Sequence of requests are ordered by timestamp, so each round would have different sequence. The first 100 rounds are only used by cloud replication algorithm as history, and the second 100 rounds are used as input sequence for all algorithms. 100 mobile nodes are simulated, and the results shown in graph are numerical average of the sum of 100 node traces, to reduce interference of random error.

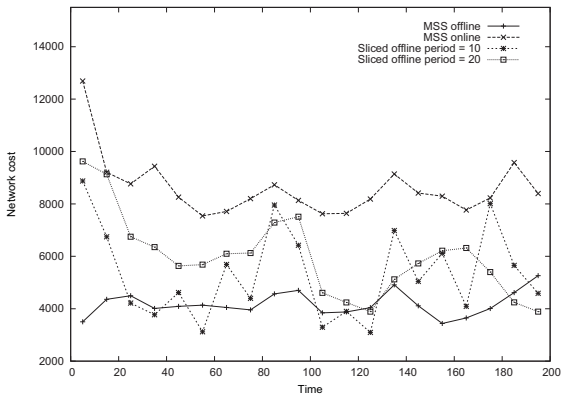

Fig. 4. Comparison the cost for various replication algorithms

Fig. 4 shows the order of performance: the best being the offline algorithm, followed by the sliced period 10, sliced period 20 and the last is online algorithm. The use of cloud enable us to use the sliced period algorithm that performs much better than the online one, which has to be used in on-premise only architectures.

We conducted another simulation to explore the tradeoff between the level of privacy protection and corresponding cost to MSS. We consider minimum privacy when every subnet edge MSS server communicates directly with the cloud. Therefore, it is possible for the cloud, by combining various information, to infer the approximate location and peers of a given subscriber, even when the IDs and addresses are

encrypted in cloud. For example, a given VIP is announced to be visiting a given city, and by analyzing the generated and received traffic of that subscriber it is possible for the cloud to infer her location in time, pattern of communications, and peers. To increase the level of privacy, we can add levels of MSS servers in the tree hybrid architecture. We assumed a hybrid tree architecture with three levels of MSS servers. Each third (edge) level sever covers a community size area of $1km^2$. Each second (middle) level sever covers a township size area of $10km^2$. Each first (top) level server covers a metropolitan size area of $100km^2$. When a third level MSS server communicates with cloud, the cloud could know that the given subscriber is in a given area of $1km^2$, which we consider minimum privacy. When the second level and first level of MSS servers communicate with the cloud, the cloud could know the location of the subscriber within a range area of $10km^2$ and $100km^2$, which we consider medium and maximum privacies respectively. But the increase in the level of privacy comes with the cost of extra MSS servers (levels of tree in the hybrid architecture and some extra network cost due to replication inside the tree structure, as shown in Fig. 5. As a consequence, users requesting higher privacy would be expected to pay more to justify the corresponding increase in service cost.

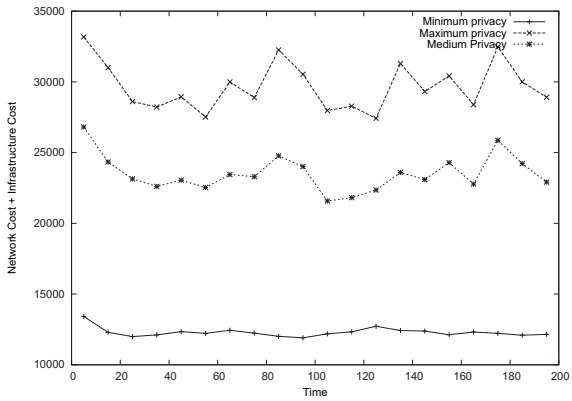

Fig. 5. Cost versus privacy level in hybrid tree architecture

VII. Conclusions

We proposed a cloud aided system for Mobility Support Service (MSS) in the Internet. MSS is offered as a value-added service, and does not require changes on access and network infrastructure. We proposed to design our mobility management system using cloud computing, which enables to leverage many economic advantages of cloud computing. Furthermore, we explore the architectural tradeoffs among service QoS, economic viability, security and privacy of various cloud aided designs. Our simulation results show how various system architectures could be used to satisfy different requirements.

Acknowledgments

This work was supported by the National Science Foundation under Grants No. 1019120, 1019119, 1138659, 1249678.

References

[1] Cisco, "Cisco Visual Networking Index: Forecast and Methodology, 2009-2014." available at http://www.cisco.com, June 2010.
[2] M. Stanley, "Internet Trends." Lecture to the National Academy of Sciences, 2010.
[3] T. R. Henderson, "Host mobility for IP networks: a comparison," *IEEE Network*, vol. 17, pp. 18–26, November 2003.
[4] E. Perera, V. Sivaraman, and A. Seneviratne, "Survey on network mobility support," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 8, no. 2, pp. 7–19, 2004.
[5] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana, "Internet indirection infrastructure," in *SIGCOMM '02: Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, (New York, NY, USA), pp. 73–86, ACM, 2002.
[6] R. Moskowitz, P. Nikander, P. Jokela, and T. Henderson, "Host Identity Protocol." RFC 5201 (Experimental), Apr. 2008.
[7] D. Farinacci, V. Fuller, D. Meyer, and D. Lewis, "Locator/id separation protocol (lisp) draft-ietf-lisp-03.txt," July 2009.
[8] C. Perkins, "IP Mobility Support for IPv4." RFC 3344 (Proposed Standard), Aug. 2002. Updated by RFC 4721.
[9] M. Buddhikot, A. Hari, K. Singh, and S. Miller, "Mobilenat: a new technique for mobility across heterogeneous address spaces," *Mob. Netw. Appl.*, vol. 10, no. 3, pp. 289–302, 2005.
[10] P. Zhang, A. Durresi, and R. Jain, "Economically Viable Support for Internet Mobility ," in *Proceedings of the International Congerence on Communications ICC*, (Kyoto, Japan), 5-9 June 2011 2011.
[11] C. Perkins, P. Calhoun, and J. Bharatia, "Mobile IPv4 Challenge/Response Extensions (Revised)." RFC 4721 (Proposed Standard), Jan. 2007.
[12] I. Akyildiz, X. Jiang, and S. Mohanty, "A survey of mobility management in next-generation all-ip-based wireless systems," in *Wireless Communications, IEEE*, vol. 11, pp. 16– 28, August 2004.
[13] J. Kempf, "Goals for Network-Based Localized Mobility Management (NETLMM)." RFC 4831 (Informational), Apr. 2007.
[14] J. Laganier, T. Koponen, and L. Eggert, "Host Identity Protocol (HIP) Registration Extension." RFC 5203 (Experimental), Apr. 2008.
[15] P. Nikander and J. Laganier, "Host Identity Protocol (HIP) Domain Name System (DNS) Extensions." RFC 5205 (Experimental), Apr. 2008.
[16] J. Day, *Patterns in Network Architecture: A Return to Fundamentals*. Prentice Hall, December 2007.
[17] J. Laganier and L. Eggert, "Host Identity Protocol (HIP) Rendezvous Extension." RFC 5204 (Experimental), Apr. 2008.
[18] B. Quoitin, L. Iannone, C. de Launois, and O. Bonaventure, "Evaluating the benefits of the locator/identifier separation," in *MobiArch '07: Proceedings of first ACM/IEEE international workshop on Mobility in the evolving internet architecture*, (New York, NY, USA), pp. 1–6, ACM, 2007.
[19] D. Andersen, "Improving end-to-end availability using overlay networks," 2005.
[20] S. Zhuang, K. Lai, I. Stoica, R. Katz, and S. Shenker, "Host mobility using an internet indirection infrastructure," *Wirel. Netw.*, vol. 11, no. 6, pp. 741–756, 2005.
[21] S. Guha and P. Francis, "An end-middle-end approach to connection establishment," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, pp. 193–204, 2007.
[22] Y. Mao, B. Knutsson, H. Lu, and J. M. Smit, "Dharma: Distributed home agent for robust mobile access," in *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies 2005 (INFOCOM 2005)*, vol. 2, pp. 1196–1206, 2005.
[23] O. Wolfson, S. Jajodia, and Y. Huang, "An adaptive data replication algorithm," *ACM Transactions on Database Systems*, vol. 22, pp. 255–314, June 1997.
[24] K. Q. Tian and D. C. Cox, *Mobility Management In Wireless Network: Data replication strategies and applications*. Boston: Kluwer Academic Publishers, December 2004.
[25] D. Neumann, M. Baker, J. Altmann, and O. F. Rana, eds., *Economic Models and Algorithms for Distributed Systems*. P.O. Box 133, CH-4010 Basel, Switzerland: Birkhauser, 2010.
[26] C. S. Alliance, "Top threats to cloud computing v1.0," March 2010.
[27] J. Winick, C. Jin, Q. Chen, and S. Jamin, "Inet Topology Generator, availavle at http://topology.eecs.umich.edu/inet/."