Unclouded Vision

Jon Crowcroft[1], Anil Madhavapeddy[1], Malte Schwarzkopf[1], Theodore Hong[1], and Richard Mortier[2]

[1] Cambridge University Computer Laboratory, 15, JJ Thomson Avenue, Cambridge CB3 0FB. UK.
firstname.lastname@cl.cam.ac.uk
[2] Horizon Digital Economy Research, University of Nottingham, Triumph Road, Nottingham NG7 2TU. UK.
firstname.lastname@nottingham.ac.uk

Abstract. Current opinion and debate surrounding the capabilities and use of the Cloud is particularly strident. By contrast, the academic community has long pursued completely decentralised approaches to service provision. In this paper we contrast these two extremes, and propose an architecture, *Droplets*, that enables a controlled trade-off between the costs and benefits of each. We also provide indications of implementation technologies and three simple sample applications that substantially benefit by exploiting these trade-offs.

1 Introduction

The commercial reality of the Internet and mobile access to it is muddy. Generalising, we have a set of cloud service providers (e.g. Amazon, Facebook, Flickr, Google and Microsoft, to name a representative few), and a set of devices that many – and soon most – people use to access these resources (so-called smartphones, e.g., Blackberry, iPhone, Maemo, Android devices). This combination of hosted services and smart access devices is what many people refer to as "The Cloud" and is what makes it so pervasive.

But this situation is not entirely new. Once upon a time, looking as far back as the 1970s, we had "thin clients" such as ultra-thin glass `tty`s accessing timesharing systems. Subsequently, the notion of thin client has periodically resurfaced in various guises such as the X-Terminal, and Virtual Networked Computing (VNC) [14]. Although the world is not quite the same now as back in those thin client days, it does seem similar in economic terms.

But why *is* it not the same? Why *should* it not be the same? The short answer is that the end user, whether in their home or on the top of the Clapham Omnibus,[3] has in their pocket a device with vastly more resource than a mainframe of the 1970s by any measure, whether processing speed, storage capacity or network access rate. With this much power at our fingertips, we should be able to do something smarter than simply using our devices as vastly over-specified dumb terminals.

[3] http://en.wikipedia.org/wiki/The_man_on_the_Clapham_omnibus

Meanwhile, the academic reality is that many people have been working at the opposite extreme from this commercial reality, trying to build "ultra-distributed" systems, such as peer-to-peer file sharing, swarms,[4] ad-hoc mesh networks, mobile decentralised social networks,[5] in complete contrast to the centralisation trends of the commercial world. We choose to coin the name "The Mist" for these latter systems.

The defining characteristic of the Mist is that data is dispersed among a multitude of responsible entities (typically, though not exclusively, ordinary users), rather than being under the control of a single monolithic provider. Haggle [17], Mirage [11] and Nimbus [15] are examples of architectures for, respectively, the networking, operating system and storage components of the Mist.

The Cloud and the Mist are extreme points in a spectrum, each with its upsides and downsides. Following a discussion of users' incentives (§2), we will expand on the capabilities of two instances of these ends later (§3). We will then describe our proposed architecture (§4) and discuss its implications for three particular application domains (§5), before concluding (§6).

2 User Incentives

For the average user, accustomed to doing plain old storage and computation on their own personal computer or mobile (what we might term "The Puddle"), there are multiple competing incentives pushing in many directions: both towards and away from the Cloud, and towards and away from the Mist (see Figure 1).

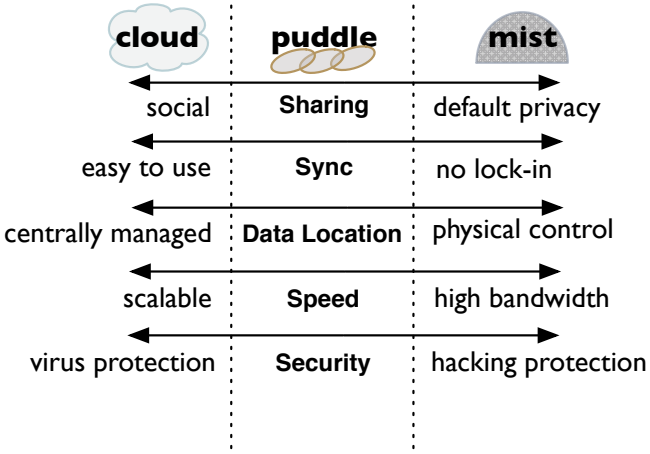

Fig. 1. Incentives pushing users toward the centralised Cloud *vs.* the decentralised Mist.

[4] `http://bittorrent.com/`
[5] `http://joindiaspora.com`, `http://peerson.net/`

Consider some of the forms of utility a user wants from their personal data:

- *Sharing.* There is a tension between the desire to share some personal data easily with selected peers (or even publicly), and the need for control over more sensitive information. The social Cloud tends to share data, whereas the decentralised Mist defaults to privacy at the cost of making social sharing more difficult.
- *Synchronization.* The Cloud provides a centralised naming and storage service to which all other devices can point. As a downside, this service typically incurs an ongoing subscription charge while remaining vulnerable to the provider stopping the service. Mist devices work in a peer-to-peer fashion which avoids provider lock-in, but have to deal with synchronisation complexity.
- *Data Location.* The Cloud provides a convenient, logically centralised data storage point, but the specific location of any component is hard for the data owner to control.[6] In contrast, the decentralised Mist permits physical control over where the devices are but makes it hard to reliably ascertain how robustly stored and backed-up the data is.
- *Speed.* A user must access a centralised Cloud via the Internet, which limits access speeds and creates high costs for copying large amounts of data. In the Mist, devices are physically local and hence have higher bandwidth. However, Cloud providers can typically scale their service much better than individuals for those occasions when "flash traffic" drives a global audience to popular content.
- *Security.* A user of the Mist is responsible for keeping their devices updated and can be vulnerable to malicious malware if they fall behind. However, the damage of intrusion is limited only to their devices. In contrast, a Cloud service is usually protected by dedicated staff and systems, but presents a valuable hacking target in which any failures can have widespread consequences, exposing the personal data of millions of users.

These examples demonstrate the clear tension between what users want from services managing their personal data *vs.* how Cloud providers operate in order to keep the system economically viable. Ideally, the user would like to keep their personal data completely private while still hosting it on the Cloud. On the other hand, the cloud provider needs to recoup hosting costs by, e.g., selling advertising against users' personal data. Even nominally altruistic Mist networks need incentives to keep them going: e.g., in BitTorrent it was recently shown that a large fraction of the published content is driven by profit-making companies rather than altruistic amateur filesharers [2].

Rather than viewing this as a zero-sum conflict between users and providers, we seek to leverage the smart capabilities of our devices to provide happy compromises that can satisfy the needs of all parties. By looking more closely at the true underlying interests of the different sides, we can often discover solutions that achieve seemingly incompatible goals [6].

[6] `http://articles.latimes.com/2010/jul/24/business/la-fi-google-la-20100724`

3 The Cloud *vs.* the Mist

To motivate the droplets architecture, we first examine the the pros and cons of the Cloud and the Mist in more detail.

The Cloud's Benefits: Centralising resources brings several significant benefits, specifically:

- economies of scale,
- reduction in operational complexity, and
- commercial gain.

Perhaps the most significant of these is the offloading of the configuration and management burden traditionally imposed by computer systems of all kinds. Additionally, cloud services are commonly implemented using virtualisation technology which enables statistical multiplexing and greater efficiencies of scale while still retaining "Chinese walls" that protect users from one another.

As cloud services have grown, they have constructed specialised technology dedicated to the task of large data storage and retrieval, for example the new crop of "NoSQL" databases in recent years [10]. Most crucially, centralised cloud services have built up valuable databases of information that did not previously exist before. Facebook's "social graph" contains detailed information on the interactions of hundreds of millions of individuals every day, including private messages and media. These databases are not only commercially valuable in themselves, they can also reinforce a monopoly position, as the network effect of having sole access to this data can prevent other entrants from constructing similar databases.

The Cloud's Costs: Why should we trust a cloud provider with our personal data? There are many ways in which they might abuse that trust, data protection legislation notwithstanding. The waters are further muddied by the various commercial terms and conditions to which users initially sign up, but which providers often evolve over time. When was the last time *you* checked the URL to which your providers post alterations to their terms and conditions, privacy policies, etc.? Even if you object to a change, can you get your data back and move it to another provider, and ensure that they have *really* deleted it?

The Mist's Benefits: Accessing the Cloud can be financially costly due to the need for constant high-bandwidth access. Using the Mist, we can reduce our access costs because data is stored (cached) locally and need only be uploaded to others selectively and intermittently. We keep control over privacy, choosing exactly what to share with whom and when. We also have better access to our data: we retain control over the interfaces used to access it; we are immune to service disruptions which might affect the network or cloud provider; and we cannot be locked out from our own data by a cloud provider.

The Mist's Costs: Ensuring reliability and availability in a distributed decentralised system is extremely complex. In particular, a new vector for breach of personal data is introduced: we might leave our fancy device on top of the aforesaid Clapham Omnibus with our data in it! We have to manage the operation of the system ourselves, and need to be connected often enough for others to be able to contact us.

Droplets: A Happy Compromise? In between these two extremes should lie the makings of a design that has all the positives and none of the negatives. In fact, a hint of a way forward is contained in the comments above.

If data is encrypted on both our personal computer/device and in the Cloud, then for privacy purposes it doesn't really matter where it is physically stored. However, for performance reasons, we do care. Hence we'd like to carry information of immediate value close to us. We would also like it replicated in multiple places for reliability reasons. We also observe that the vast majority of user-generated content is of interest *only* within the small social circle of the content's subject/creator/producer/owner and thus note that interest/popularity in objects tends to be Zipf-distributed.

In the last paragraph, it might be unclear who "we" are: "we" refers to Joe Public, whether sitting at home or on the top of that bus. However, there is another important set of stakeholders: those who provide The Cloud and The Net. These stakeholders need to make money lest all of this fail.

The service provider needs revenue to cover operational expenses and to make a profit, but is loath to charge the user directly. Even in the case of the network, ISPs (and 3G providers) are mostly heading toward flat data rates. As well as targeted advertisements and associated "click-through" revenue, service providers also want to carry out data mining to do market research of a more general kind.

Fortunately, recent advances in cryptography and security hint at ways to continue to support the two-sided business models that abound in today's Internet. In the case of advertising, the underlying interest of the Cloud provider is actually the ability to sell targeted ads, not to know everything about its users. Privacy-preserving query techniques can permit ads to be delivered to users matching certain criteria without the provider actually knowing which users they were [8, 9, 16]. In the case of data mining on the locations or transactions of users, techniques such as differential privacy [5] and k-anonymity [18] can allow providers to make queries on aggregate data without being able to determine information about specific users.

So we propose *Droplets*, half way between the Cloud and the Mist. Droplets make use of the Mirage operating system [11], Nimbus storage [15] and Haggle networking [17]. They float between the personal device and the cloud, using technologies such as social networks, virtualisation and migration [1, 3], and they provide the basic components of a Personal Container [12]. They condense within social networks, where privacy is assured by society, but in the great unwashed Internet, they stay opaque. The techniques referred to above allow the service

providers to continue to provide the storage, computation, indexing, search and transmission services that they do today, with the same wide range of business models.

4 Droplets

Droplets are units of network-connected computation and storage, designed to migrate around the Internet and personal devices. At a droplet's core is the *Mirage* operating system, which compiles high-level language code into specialised targets such as Xen micro-kernels, UNIX binaries, or even Javascript applications. The same Mirage source code can thus run on a cloud computing platform, within a user's web browser, on a smart-phone, or even as a plugin on a social network's own servers. As we note in Table 1, there is no single "perfect" location where a Droplet should run all the time, and so this agility of placement is crucial to maximising satisfaction of the users' needs while minimising their costs and risks.

Platform	*Google AppEngine*	*VM (e.g., on EC2)*	*Home Computer*	*Mobile Phone*
Storage	moderate	moderate	high	low
Bandwidth	high	high	limited	low
Accessibility	always on	always on	variable	variable
Computation	limited	flexible, plentiful	flexible, limited	limited
Cost	free	expensive	cheap	cheap
Reliability	high	high	medium (failure)	low (loss)

Table 1. Comparison of different potential Droplets platforms.

Storage in such an environment presents a notable challenge, which we address via the *Nimbus* system, a distributed, encrypted and delay-tolerant personal data store. Working on the assumption that personal data access follows a Zipf power-law distribution, popular objects can be kept live on relatively expensive but low-latency platforms such as a Cloud virtual machine, while older objects can be archived inexpensively but safely on a storage device at home. Nimbus also provides local attestation in the form of "trust fountains," which let nodes provide a cryptographic attestation witnessing another node's presence or ownership of some data. Trust fountains are entirely peer-to-peer, and so proof is established socially (similarly to the use of lawyers or public notaries) rather than via central authority.

Haggle provides a delay-tolerant networking platform, in which all nodes are mobile and can relay messages via various routes. Even with the use of central "stable" nodes such as the Cloud, outages will still occur due to the scale and

dynamics of the Cloud and the Net, as has happened several times to such high-profile and normally robust services as GMail. During such events, the user must not lose all access to their data, and so the Haggle delay-tolerant model is a good fit. It is also interesting to observe that many operations performed by users are quite latency-insensitive, e.g. backups can be performed incrementally, possibly overnight.

4.1 Deployment Model

Droplets are a compromise between the extremely-distributed Mist model and the more centralised Cloud. They store a user's data and provide a network interface to this data rather than exposing it directly. The nature of this access depends on where the Droplet has condensed.

- *Internet droplet.* If the Droplet is running exposed to the wild Internet, then the network interfaces are kept low-bandwidth and encrypted by default. To prevent large-scale data leaks, the Droplet rejects operations that would download or erase a large body of data.
- *Social network droplet.* For hosting data, a droplet can condense directly within a social network, where it provides access to its database to the network, e.g., for data mining, in return for "free" hosting. Rather than allowing raw access, it can be configured to only permit aggregate queries to help populate the provider's larger database, but still keep track of its own data.
- *Mobile droplet.* The Droplet provides high-bandwidth, unfettered access to data. It also regularly checks with any known peers to see if a remote wipe instruction will cause it to permanently stop serving data.
- *Archiver droplet.* Usually runs on a low-power device, e.g., an ARM-based BeagleBoard, accepting streams of data changes but not itself serving data. Its resources are used to securely replicate long-term data, ensuring it remains live, and to alert the user in case of significant degradation.
- *Web droplet.* A Droplet in a web browser executes as a local Javascript application, where it can provide web bookmarklet services, e.g., trusted password storage. It uses cross-domain AJAX to update a more reliable node with pertinent data changes.

Droplets can thus adapt their external interfaces depending on where they are deployed, allowing negotiation of an acceptable compromise between hosting costs and desire for privacy.

4.2 Trust Fountains

To explain trust fountains by way of example, consider the following. As part of the instantiation of their Personal Container, Joe Public runs an instance of a Nimbus trust fountain. When creating a droplet from some data stored in his Personal Container, this trust fountain creates a cryptographic attestation

proving Joe's ownership of the data at that time in the form of a time-dependent hash token.

The droplet is then encrypted under this hash token using a fast, medium strength cipher[7] and pushed out to the cloud. By selectively publishing the token, Joe can grant access to the published droplet e.g., allowing data mining access to a provider in exchange for free data storage and hosting. Alternatively, the token might only be shared with a few friends via an *ad hoc* wireless network in a coffee shop, granting them access only to that specific data at that particular time.

4.3 Backwards Provenance

A secondary purpose of the attestation is to enable "backwards provenance", i.e., a way to prove ownership. Imagine that Joe publishes a picture of some event which he took using his smartphone while driving past it on that oft-considered bus. A large news agency picks up and uses that picture after Joe publishes it to his Twitter stream using a droplet. The attached attestations then enable the news agency to compensate both the owner and potentially the owner's access provider, who takes a share in all profits made from Joe's digital assets in exchange for serving them.

Furthermore, Joe is given a tool to counter "hijacking" of his creation even if the access token becomes publicly known: using the cryptographic properties of the token, the issue log of his trust fountain together with his provider's confirmation of receipt of the attested droplet forms sufficient evidence to prove ownership and take appropriate legal action. Note that Joe Public can also deny ownership if he chooses, as only his trust fountain holds the crucial information necessary to regenerate the hash token and thus prove the attestation's origin.

4.4 Handling 15 Minutes of Fame

Of course, whenever a droplet becomes sufficiently popular to merit condensation into a cloud burst of marketing, then we have the means to support this transition, and we have the motivation and incentives to make sure the right parties are rewarded. In this last paragraph, "we" refers to all stakeholders: users, government and business. It seems clear that the always-on, everywhere-logged, ubiquitously-connected vision will continue to be built, while real people become increasingly concerned about their privacy [4]. Without such privacy features, it is unclear for how much longer the commercial exploitation of personal data will continue to be acceptable to the public; but without such exploitation, it is unclear how service providers can continue to provide the many "free" Internet services on which we have come to rely.

[7] Strong encryption is not required as the attestations are unique for each droplet publication and breaking one does not grant an attacker access to any other droplets.

5 Droplications

The Droplet model requires us to rethink how we construct applications – rather than building centralised services, they must now be built according to a distributed, delay-tolerant model. In this section, we discuss some of the early services we are building.

5.1 Digital Yurts

In the youthful days of the Internet, there was a clear division between public data (web homepages, FTP sites, etc.) and private (e-mail, personal documents, etc.). It was common to archive personal e-mail, home directories and so on, and thus to keep a simple history of all our digital activities. The pace of change in recent years has been tremendous, not only in the *variety* of personal data, but in *where* that data is held. It has moved out of the confines of desktop computers to data-centres hosted by third-parties such as Google, Yahoo and Facebook, who provide "free" hosting of data in return for mining information from millions of users to power advertising platforms.

These sites are undeniably useful, and hundreds of millions of users voluntarily surrender private data in order to easily share information with their circle of friends. Hence, the variety of personal data available online is booming – from media (photographs, videos), to editorial (blogging, status updates), and streaming (location, activity).

However, privacy is rapidly rising up the agenda as companies such as Facebook and Google collect vast amounts of data from hundreds of millions of users. Unfortunately, the only alternative that privacy-sensitive users currently have is to delete their online accounts, losing both access to and what little control they have over their online social networks. Often, deletion does not even completely remove their online presence. We have become digital nomads: we have to fetch data from many third-party hosted sites to recover a complete view of our online presence. Why is it so difficult to go back to managing our own information, using our own resources? Can we do so while keeping the "good bits" of existing shared systems, such as ease-of-use, serendipity and aggregation?

Although the immediate desire to regain control of our privacy is a key driver, there are several other longer-term concerns about third parties controlling our data. The incentives of hosting providers are not aligned with the individual: we care about preserving our history over our lifetime, whereas the provider will choose to discard information when it ceases to be useful for advertising.

This is where the Droplet model is useful – rather than dumbly storing data, we can also negotiate access to that data with hosting providers via an Internet droplet, and arrive at a compromise between letting them data mine it, versus the costs of hosting it. When the hosting provider loses interest in the older, historical tail of data, the user can deploy an archival droplet to catch the data before it disappears, and archive it for later retrieval.

5.2 Dust Clouds

Dust Clouds [13] is a proposal for the provision of secure anonymous services using extremely lightweight virtual machines hosted in the cloud. As they are lightweight, they can be created and destroyed with very short lifetimes, yet still achieve useful work. However, several tensions exist between the requirements of users and cloud providers in such a system.

For example, cloud providers have a strong requirement for a variety of auditing functions. They need to know who consumed what resources in order to bill, to provision appropriately, to ensure that, e.g., upstream service level agreements with other providers are met, and so on. They would tend to prefer centralisation for reasons already mentioned (efficiency, economy of scale, etc.).

By contrast, individual consumers use such a system precisely because it provides anonymity while they are doing things that they wish not to be attributed to them, e.g., to avoid arrest. Anonymity in a dust cloud is largely provided by having a rich mixnet of traffic and other resource consumption. Consumers would also prefer diversity, in both geography and provider, to ensure that they're not at the mercy of a single judicial/regulatory system.

Pure cloud approaches fail the users' requirements by putting too much control in the hands of one (or a very small number of) cloud providers. Pure mist approaches fail the user by being unable to provide the richness of mixing to provide sufficient anonymity: many of the devices in the mist are either insufficiently powerful or insufficiently well-connected to support a large enough number of users' processes. By taking a droplets approach we obviate both these issues: the lightweight nature of VM provisioning means that it becomes largely infeasible for the cloud provider to track in detail what users are doing, particularly when critical parts of the overall distributed process/communication are hosted on non-cloud infrastructure. Local auditing for payment recovery based on resources used is still possible, but the detailed correlation and reconstruction of an individual process's behaviour becomes effectively impossible. At the same time, the scalable and generally efficient nature of cloud-hosted resources can be leveraged to ensure that the end result is itself suitably scalable.

5.3 Evaporating Droplets

With Droplets, we also have a way of creating truly ephemeral data items in a partially trusted or untrusted environment, such as a social network, or the whole Internet. Since Droplets have the ability to do computation, they can refuse to serve data if access prerequisites are not met: for example, time-dependent hashes created from a key and a time stamp can be used to control access to data in a Droplet. Periodically, the user's "trust fountain" will issue new keys, notifying the Droplet that it should now accept the new key only. To "evaporate" data in a Droplet, the trust fountain simply ceases to provide keys for it, thus making users unable to access the Droplet, even if they still have the binary data or even the Droplet itself (assuming, of course, that brute-forcing the hash key is not a worthwhile option). Furthermore, their access is revoked even in disconnected

state, i.e. when the Droplet cannot be notified to accept the new hash key only: since it is necessary to provide time stamp and key as authentication tokens in order for the Droplet to generate the correct hash, expired keys can no longer be used as they have to be provided along with their genuine origin time stamp. Additionally, as a more secure approach, the Droplet could even periodically re-encrypt its contents in order to combat brute-forcing.

This subject has been of some research interest recently. Another approach [7] relies on statistical metrics that require increasingly large amounts of data from a DHT to be available to an attacker in order to reconstruct the data, but is vulnerable to certain Sybil attacks [19]. Droplets, however, have the power of being able to completely ensure that all access to data is revoked, even when facing a powerful adversary in a targeted attack.

Furthermore, as a side effect of the hash-key based access control, the evaporating Droplet could serve different views, or stages of evaporation, to different requesters depending on the access key they use (or its age).

Finally, the "evaporating Droplet" can be made highly accessible from a user perspective by utilizing a second Droplet: a Web Droplet (see §4.1) that integrates with a browser can automate the process of requesting access keys from trust fountains and unlocking the evaporating Droplet's contents.

6 Conclusions and Future Work

In this paper, we have discussed the tension between the capabilities of and demands on the Cloud and the Mist. We concluded that both systems are at opposite ends of a spectrum of possibilities and that compromise between providers and users is essential. From this, we derived an architecture for an alternative system, *Droplets*, that enables control over the trade-offs involved, resulting in systems acceptable to both hosting providers and users.

Having realised two of the main components involved in Droplets, Haggle networking and the Mirage operating system, we are now completing realisation of the third, Nimbus storage, as well as building some early "droplications".

References

1. Clark, C., Fraser, K., Hand, S., Hansen, J.G., Jul, E., Limpach, C., Pratt, I., Warfield, A.: Live migration of virtual machines. In: USENIX Symposium on Networked Systems Design & Implementation (NSDI). pp. 273–286. USENIX Association, Berkeley, CA, USA (2005)
2. Cuevas, R., Kryczka, M., Cuevas, A., Kaune, S., Guerrero, C., Rejaie, R.: Is content publishing in BitTorrent altruistic or profit-driven (Jul 2010), `http://arxiv.org/abs/1007.2327`
3. Cully, B., Lefebvre, G., Meyer, D.T., Karollil, A., Feeley, M.J., Hutchinson, N.C., Warfield, A.: Remus: High availability via asynchronous virtual machine replication. In: USENIX Symposium on Networked Systems Design & Implementation (NSDI). USENIX Association, Berkeley, CA, USA (April 2008)

4. Doctorow, C.: The Things that Make Me Weak and Strange Get Engineered Away. Tor.com (August 2008), http://www.tor.com/stories/2008/08/weak-and-strange
5. Dwork, C.: Differential privacy. In: International Colloquium on Automata, Languages and Programming (ICALP). pp. 1–12 (2006)
6. Fisher, R., Patton, B.M., Ury, W.L.: Getting to Yes: Negotiating Agreement Without Giving In. Houghton Mifflin (April 1992), http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20\&path=ASIN/0395631246
7. Geambasu, R., Kohno, T., Levy, A., Levy, H.M.: Vanish: Increasing data privacy with self-destructing data. In: Proceedings of the USENIX Security Symposium (August 2009)
8. Guha, S., Reznichenko, A., Tang, K., Haddadi, H., Francis, P.: Serving Ads from localhost for Performance, Privacy, and Profit. In: Proceedings of Hot Topics in Networking (HotNets). New York, NY (Oct 2009)
9. Haddadi, H., Hui, P., Brown, I.: MobiAd: Private and scalable mobile advertising. In: Proceedings of MobiArch 2010 (2010), to appear.
10. Leavitt, N.: Will nosql databases live up to their promise? Computer 43(2), 12–14 (February 2010), http://dx.doi.org/10.1109/MC.2010.58
11. Madhavapeddy, A., Mortier, R., Sohan, R., Gazagnaire, T., Hand, S., Deegan, T., McAuley, D., Crowcroft, J.: Turning down the LAMP: software specialisation for the cloud. In: HotCloud'10: Proceedings of the 2nd USENIX conference on Hot topics in cloud computing. pp. 11–11. USENIX Association, Berkeley, CA, USA (2010)
12. Mortier, R., *et al.*: The Personal Container, or Your Life In Bits (October 2010), proceedings of Digital Futures
13. Mortier, R., Madhavapeddy, A., Hong, T., Murray, D., Schwarzkopf, M.: Using Dust Clouds to enhance anonymous communication. In: Proceedings of the Eighteenth International Workshop on Security Protocols (IWSP) (April 2010)
14. Richardson, T., Stafford-Fraser, Q., Wood, K.R., Hopper, A.: Virtual network computing. IEEE Internet Computing 2(1), 33–38 (January 1998)
15. Schwarzkopf, M., Hand, S.: Nimbus: Intelligent Personal Storage (2010), poster at the Microsoft Research Summer School 2010, Cambridge, UK.
16. Shikfa, A., Önen, M., Molva, R.: Privacy in content-based opportunistic networks. In: AINA Workshops. pp. 832–837 (2009)
17. Su, J., Scott, J., Hui, P., Crowcroft, J., De Lara, E., Diot, C., Goel, A., Lim, M.H., Upton, E.: Haggle: seamless networking for mobile applications. In: UbiComp '07: Proceedings of the 9th international conference on Ubiquitous computing. pp. 391–408. Springer-Verlag, Berlin, Heidelberg (2007)
18. Sweeney, L.: k-anonymity: a model for protecting privacy. Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 10(5), 557–570 (2002)
19. Wolchok, S., Hofmann, O., Heninger, N., Felten, E., Halderman, J., Rossbach, C., Waters, B., Witchel, E.: Defeating vanish with low-cost sybil attacks against large DHTs. In: Proceedings of the 17th Network and Distributed System Security Symposium (NDSS). pp. 37–51 (2010)