The Role of Cloudlets in Hostile Environments

The convergence of mobile computing and cloud computing is predicated on a reliable, high-bandwidth, end-to-end network, which is difficult to guarantee in hostile environments. However, virtual-machine-based cloudlets located in close proximity to associated mobile devices can overcome this deep-rooted problem.

Mahadev Satyanarayanan
Carnegie Mellon University

Grace Lewis, Edwin Morris, Soumya Simanta, and Jeff Boleng
Software Engineering Institute

Kiryong Ha
Carnegie Mellon University

The explosive growth of mobile computing over the last decade has been driven by consumer demand for smartphones, tablets, and other mobile devices. Today, these devices and their associated software operate in a stable Internet setting with 3G, 4G, or Wi-Fi last-hop connectivity. In this familiar world, the convergence of mobile and cloud computing is well under way. Implicit in this convergence is the assumption that the cloud is easily accessible at all times. In other words, there's good end-to-end network quality and few network or cloud failures.

Here, we examine a very different world: one in which connectivity to the cloud can't be taken for granted. In this world, cloud access must be viewed as a luxury rather than a birthright. This viewpoint applies to several important contexts that we collectively refer to as *hostile environments*. The prime example of a hostile environment is a theater of military operations. Another example is a geographical region where recovery is underway after a natural disaster or terrorist attack. A third example is a developing country with a weak networking infrastructure.

More generally, even a well-connected region of the public Internet can become a hostile environment during a cyberattack. The volume of cyberattacks in the past few years confirms that this isn't just a hypothetical possibility. There's growing concern that cyberattacks might soon become major weapons of organized crime as well as instruments of national policy.[1] If these dire predictions come true, we might have no choice but to view the entire wide-area Internet as a hostile environment. Two recent events offer a foretaste of the pain experienced by users when a cloud-based service becomes unavailable: the day-long outage of Apple's Siri voice recognition system in November 2011,[2] and the extended Christmas Eve 2012 outage of Netflix's video streaming service due to an Amazon failure.[3]

Cloudlets, originally motivated by narrow considerations of end-to-end latency,[4] can play a much broader and more foundational role for mobile computing in hostile environments. A cloudlet can be viewed as a surrogate or proxy of the real cloud, located as the middle tier of a three-tier hierarchy: mobile device, cloudlet, and cloud. We advocate a design strategy in which a cloudlet is completely transparent under normal conditions, giving mobile users the illusion that they're directly interacting with the cloud. Under failure conditions, the cloudlet masks the absence of the cloud by performing its essential services.

Taxonomy of Cloud Usage

To understand the impact of hostile environments, we first must examine the different ways in

Published by the IEEE CS ■ 1536-1268/13/$31.00 © 2013 IEEE

which the cloud can be leveraged to improve mobile computing.

Cloud Benefits

Mobile users can leverage the cloud in at least three distinct ways.

Overcoming resource limitations. As mobile computing pushes beyond today's familiar desktop, laptop, and smartphone applications to capabilities such as free-form speech recognition, natural language translation, face and object recognition, dynamic activity interpretation from video, and body language interpretation, it becomes necessary to amplify the compute power of resource-challenged mobile devices. Extending battery life, speeding up execution, and solving larger problems can all be obtained by offloading resource-intensive computation to the cloud. Jason Flinn gives an excellent review of the extensive work in this space over the past decade.[5]

Authoritative sourcing of data. The cloud serves as the custodian of the definitive versions of all shared data. Many local copies (such as cache copies or replicas) of a data item can be created for performance or reliability reasons, but the cloud is always the definitive source. The nature of the data, the protocol to access it from the cloud, and other attributes can vary widely across collections of data items—for example, files in a distributed file system, such as the Andrew File System (AFS) or Coda; streamed video from YouTube; map data from Google Maps; or HTML pages from a Web server. Although this use of the term "cloud" is new, the abstraction of a single authoritative system-wide data source goes back to AFS in Project Andrew[6,7] in the mid-1980s.

Synchronizing multi-user collaboration. Through a variety of workflow-related software, the asynchronous actions of a group of mobile users can be sequenced correctly. The definition of "correct" is, of course, workflow-specific. The cloud acts as the hub of collaboration, even though the participation of individual users might be asynchronous with respect to space and time.

Computer-supported cooperative work software, also known as "groupware," is an obvious example of this genre of cloud usage. Another example is commercial conferencing software, such as WebEx, GoToMeeting, and Skype. Many aspects of business process software (such as PeopleSoft and SAP) also use the cloud in this way. Enabling a team of dispersed mobile users to collaborate on a task or mission is highly valuable in many hostile environments.

Cloud Disruptions

An inability to access the cloud is disruptive in all three cases. In the first case (cyber foraging), cloud-dependent mobile applications will be inoperable or will resort to degraded local fallbacks that offer a poor user experience, such as lower fidelity, sluggish or jerky response, or shortened battery life.

In the second case (data access), critical data items might become unreadable, stale, inconsistent, or unmodifiable.

In the third case (collaboration), the correctness criteria for specific workflows might fail to be enforced. Alternatively, a group of collocated users might be prevented from collaborating because of an overly conservative synchronization strategy.

Hostile Environments

Short-term large-magnitude uncertainty is one of the dominant attributes of hostile environments. This contrasts with the well-conditioned, low-uncertainty environment that most Internet users experience today. Note that minor failures and some "burstiness" of resource demands are already factored into the design of today's Internet applications. For example, TCP retransmission and adaptive windowing masks packet loss and network congestion. Elastic computing mechanisms within a cloud dynamically allocate virtual and physical machine resources based on current workload demands. RAID storage masks unpredictable disk failures and permits online repair. These mechanisms were conceived for benign Internet environments in which failures and overloads were random natural events rather than deliberate actions of clever adversaries.

By definition, a hostile environment overwhelms engineering practices that are adequate for coping with everyday uncertainties. Designing for a hostile environment requires addressing worst-case assumptions rather than average-case assumptions, which drive the design choices and economic models of Internet applications today.

Military Operations

Military settings represent the worst-case scenario from a survivability viewpoint. An architecture that's robust in the face of sustained adversarial attacks will likely survive the worst that Mother Nature or human folly can throw at it. As an analogy, the robustness and survivability of today's Internet is largely due to the principles of dispersion, decentralization, and layer independence that pervade its design. Those principles emerged from the military roots of the Internet and its non-negotiable mandate for survivability.

The US Department of Defense has long been a proponent of mobile computing for foot soldiers, with prototype systems such as Land Warrior dating back to the mid 1990s.[8] (See the sidebar for some illustrative use cases.) All three modes of cloud usage discussed earlier—offloading, data sourcing, and team collaboration—are relevant to military operations.

Unfortunately, the network connection to a distant cloud is vulnerable to wireless jamming or other modes of denial of service. The DoS threat can never be completely eliminated when most communication is through wireless channels. Jamming of wireless signals continues to be a threat today, in spite

Example Use Cases

Here, we present sample uses cases for both military operations and disaster recovery.

Military Operations

For our first example, consider a group of soldiers. They've just captured a person and must confirm his identity. They take his picture and compare it to a continuously updated remote image database. After finding a match—with a key enemy officer—they send the captive to an interrogation center. (This example is based on reports that face recognition technology played a pivotal role in helping Navy SEALs verify Osama bin Laden's identity in Abbottabad, Pakistan.[1])

Next, consider a forward unit that has been alerted to a possible chemical or biological attack. A soldier takes an air sample using a portable sensor that includes a mass spectrometer. Compute-intensive analysis of its output reveals the presence of a known chemical agent. Although its concentration is still below the hazard threshold, a repeat measurement indicates rising concentration. With timely warning, the unit evacuates to safety. For more information, see Brian Sullivan, Bruce Evans, and Phil Allen's overview of field detection of chemical and biological agents.[2]

Finally, imagine a soldier is trying to gather information from residents of a remote area that was recently under attack. The soldier knows that these residents speak Pashto rather than Dari or Arabic. With the appropriate smartphone settings, the soldier hears translated English from Pashto. Responses generate real-time translations in spoken Pashto. For more information, see Ehud Rattner's discussion of language translation in the field in Afghanistan.[3]

Disaster Recovery

After a massive 9.1 earthquake and resulting tsunami, disaster relief is painfully slow. First responders are guided by now-obsolete maps, surveys, photographs, and building floor plans. Major highways on their maps are no longer usable, and bridges, buildings, and landmarks have collapsed.

Consider a group of responders who, desperate to succeed in their rescue efforts, turn to an emerging technology: camera-based GigaPan sensing. Using off-the-shelf consumer-grade cameras in smartphones, local citizens take hundreds of close-up images of disaster scenes. These crowd-sourced images are stitched together into a zoomable panorama using compute-intensive vision algorithms. As new maps and topographical overlays are constructed, rescue efforts speed up and become more effective.

Karen Frenkel gives a good overview of GigaPan applications.[4] Figure A1 shows a GigaPan image of downtown Port Au Prince after the 2010 earthquake, assembled from photographs taken by a *New York Times* reporter shortly after the event. The zoomed-in view in Figure A2 identifies a specific utility pole that has been destroyed—information that could be valuable in restoring communication.

REFERENCES

1. J. Vazquez, "How Facial Recognition Technology Works," Ozarksfirst.com, May 2011; www.ozarksfirst.com/story/how-facial-recognition-technology-works/d/story/I2CvMIPsMEGKvDLjZkc13Q.
2. B.M. Sullivan, B.W. Evans, and P.W. Allen, "Biological and Chemical Warfare Defense Sensors and Systems," *Systems and Information Technology Rev. J.*, vol. 8, no. 1, 2000, pp. 103–110.
3. E. Rattner, "Afghan Language Translation Devices for U.S. Army," *The Future of Things*, Aug. 2010; http://thefutureofthings.com/news/10229/afghan-language-translation-devices-for-u-s-army.html#.
4. K.A. Frenkel, "Panning for Science," *Science*, vol. 330, no. 6005, 2010, pp. 748–749.

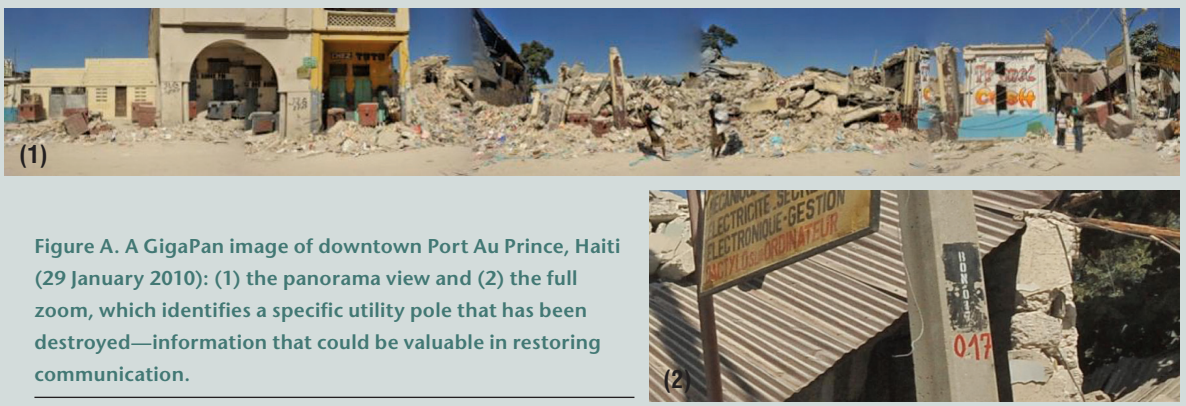


Figure A. A GigaPan image of downtown Port Au Prince, Haiti (29 January 2010): (1) the panorama view and (2) the full zoom, which identifies a specific utility pole that has been destroyed—information that could be valuable in restoring communication.

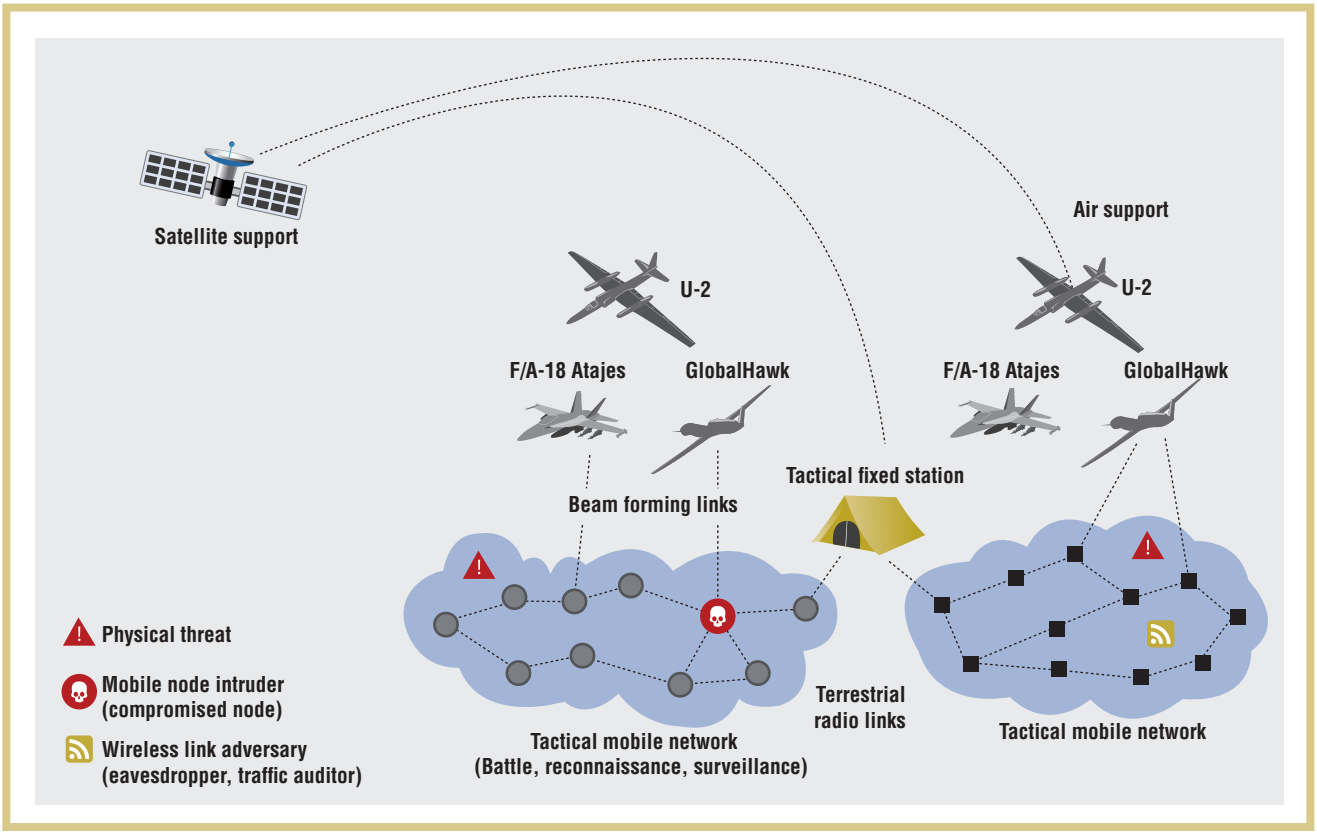


Figure 1. An example combat network.[9] The wide-area links based on satellite and air support are the most vulnerable to wireless jamming attacks from a distance.

of mechanisms such as spread-spectrum transmission and frequency-hopping. Also, there might be extended periods during which an adversary remains quiet to induce a false sense of security before disrupting network communications at a critical moment.

Figure 1 illustrates typical wireless communication links in a modern battle space.[9] Although the operational characteristics of many wireless technologies remain classified, certain broad principles can be identified from the viewpoint of DoS attacks. The wide-area links based on satellite and air support are the most vulnerable to wireless jamming attacks from a distance. The time to repair these links is lengthy, compared to repairing a typical cloud operation.

At the tactical level, ad hoc multihop networks (as shown in Figure 1) will likely be prevalent in the future. In addition to jamming, these networks are also vulnerable to unique routing-based attacks.[10] Examples include *wormholes*, in which two rogue nodes create a link with artificially good performance and then drop packets once they're adopted as a good route; and *rushing*, in which an attacker fabricates route requests that result in the network being unable to find routes longer than two hops. Depending on the wireless technology, the hop distance can vary from a meter or less to a few tens of kilometers. As a broad generalization, wireless technologies with short range tend to support higher bandwidths, are less vulnerable to jamming and less detectable from a distance, and consume less power.

Disaster Recovery

Natural disasters of monumental scale and destruction visit us with terrifying frequency: Hurricane Sandy in 2012; tsunamis in Tohoku in 2011 and in the Indian Ocean in 2004; earthquakes in Chile and Haiti in 2010, and in Sichuan in 2008; and the New Orleans Katrina flooding in 2005, to name just a few from the recent past. We can add terrorist attacks such as 9/11 to this list. The National Academies report by Ramesh Rao and his colleagues examines how information technology can improve the speed and effectiveness of disaster recovery.[11]

Sudden obsolescence of information regarding terrain and buildings is a major contributor to slow disaster response. Vital sources of knowledge—such as maps, surveys, photographs, and building floor plans—are no longer valid. Major highways on a map are no longer usable. Bridges, buildings, and landmarks have collapsed. Even the physical topography of an affected area can be severely changed.

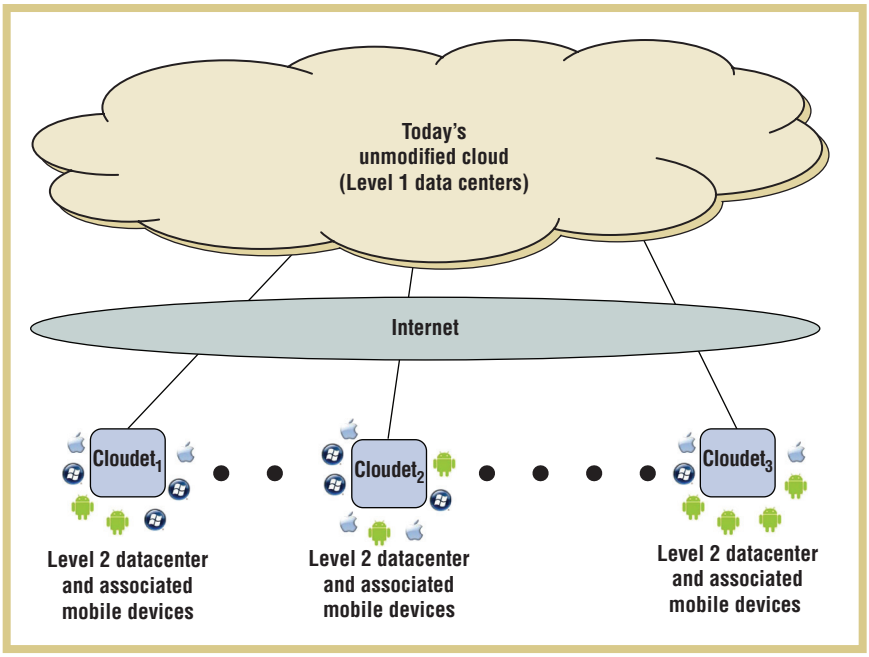


Figure 2. A two-level cloud computing architecture. Level 1 is today's unmodified cloud infrastructure, and level 2 comprises cloudlets that are dynamically associated with nearby mobile devices.

Conducting search and rescue missions in the face of obsolete information is difficult and dangerous. Relative to the taxonomy presented earlier, in this situation, the authoritative version of data is no longer in the cloud; rather, it must be recreated bottom-up at the edges (and eventually propagated to the cloud). New knowledge of terrain and buildings must be reconstructed from scratch at sufficient resolution to make important life and death decisions in search and rescue missions. As the example in the sidebar illustrates, crowd-sourced imaging and geolocation by mobile users can play a valuable role in remapping.

Poor Internet connectivity is another factor complicating relief efforts. The physical infrastructure necessary for good Internet connectivity (such as undersea cables) might have been destroyed, and it could be many days or weeks before these can be repaired. Limited Internet connectivity can be re-established soon after the catastrophic event, but there will be a high demand for this scarce resource from diverse sources: families trying to learn and share information about the fate of loved ones; citizen reporters and professional journalists sharing videos, images, blogs, and tweets of the disaster area with the outside world; and disaster relief agencies coordinating their efforts with their home bases.

Interoperability issues arising from hardware and software heterogeneity among rescue teams is another complication. After a major disaster, responding personnel come from many different organizations (and possibly many different parts of the world), each with its own equipment and software environments. The ability to coordinate missions across diverse personnel is hindered by the lack of software interoperability. This limits the sharing of relevant information and efficient use of scarce resources. The ability to borrow or share computing hardware across different groups is also severely restricted. The obvious solution of enforcing standardization at a global scale, especially among philanthropic organizations that are largely volunteer-based, is impractical.

Architectural Considerations

Why are cloudlets relevant to hostile environments? Their original motivation was to reduce end-to-end latency of cloud offload from mobile devices for applications that are both resource-intensive and latency-sensitive.[4] In hostile environments, the physical proximity of cloudlets brings additional benefits, which we discuss in the next section. Here, we first examine certain architectural issues to establish the context for that discussion.

Placing the cloud in physical proximity to mobile devices is neither feasible nor advisable on a global scale. Cloudlets thus emerge as surrogates or proxies of the cloud that embody the critical property of proximity. The challenge is to architect cloud usage in such a way that all of the usage modalities discussed earlier benefit from the associated cloudlet's proximity, but with minimal disruptions for software applications and cloud services.

This can be achieved through a hierarchical extension of today's cloud computing infrastructure, as shown in Figure 2. Level 1 of this hierarchy is an unmodified cloud infrastructure, such as Amazon's EC2 data centers. Level 2 comprises cloudlets that are dynamically associated with mobile devices in their proximity (much like Wi-Fi access points).

Many of the arguments for using virtual machines (VMs) at level 1 also apply at level 2. These include

- strong isolation between untrusted user-level computations;
- mechanisms for authentication, access control, and metering;
- dynamic resource allocation for user-level computations; and
- the ability to support a wide range of user-level computations, with minimal restrictions on their process structure, programming languages, or operating systems.

By leveraging the rich ecosystem of VM-based mechanisms, policies, and practices that already exists for level 1, we can simplify and make transparent the migration of cloud functionality between levels 1 and 2.

The architecture shown in Figure 2 encourages an appliance-like deployment model for cloudlets, with no active management after installation. Only a bare minimum of state is preinstalled on the persistent local storage of a cloudlet. Most of the cloudlet state is either cached from level 1 or regenerated locally. Examples of such state include VM images and files from a distributed file system. Because cloudlets in hostile environments might experience frequent disconnections from level 1, techniques (such as hoarding[12]) to prefetch the state before it's needed will be important. Dynamic VM synthesis makes it possible to rapidly create the missing VM state on a cloudlet,[4] even when it's disconnected from level 1.

The absence of hard state on cloudlets simplifies management. Consolidating or reconfiguring level 1 data centers doesn't affect cloudlets. Adding a new cloudlet or replacing an existing one only requires modest setup and configuration. Once installed, a cloudlet can dynamically self-provision from level 1. The physical motion of a mobile device can take it far from the cloudlet with which it's currently associated. In that case, a mechanism similar to wireless access point handoff can be executed to seamlessly switch association to a close cloudlet and migrate back-end execution state to that cloudlet.

The DoS vulnerability considerations discussed earlier, while focused on military operations, also apply to disaster recovery scenarios. In this case, the wireless backhaul is fragile because of capacity overload rather than DoS attacks. An ensemble of cloudlets with mesh connectivity, as shown in Figure 3, can offer reliable cloud computing services within a disaster recovery area, even with a fragile backhaul to the cloud.

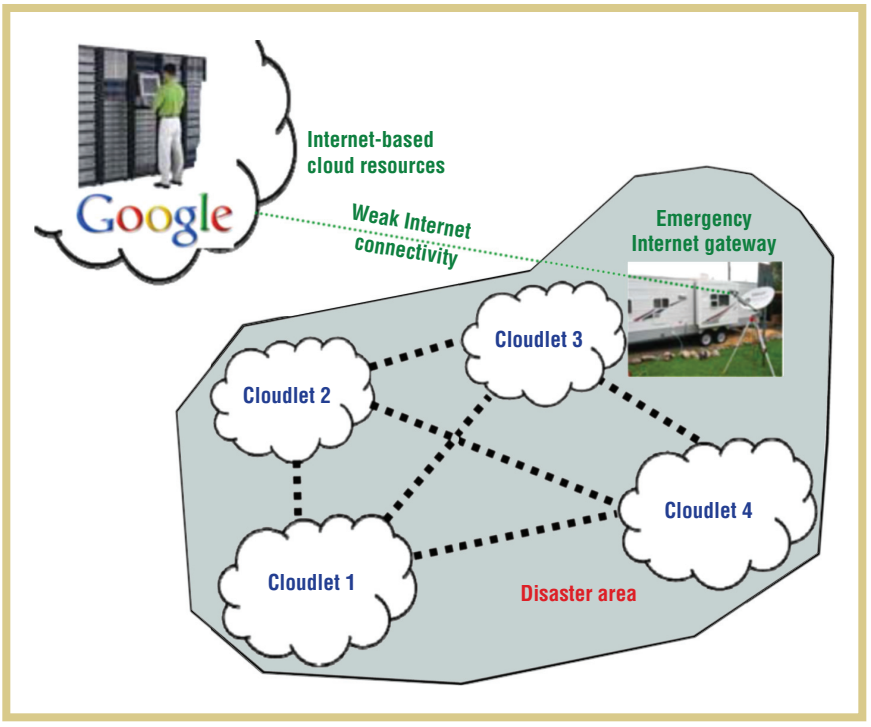


Figure 3. A cloudlet ensemble. An ensemble of cloudlets with mesh connectivity can offer reliable cloud computing services within a disaster recovery area even with a fragile backhaul to the cloud.

How Cloudlets Can Help

A cloudlet can be viewed as a "data center in a box" that "brings the cloud closer."[4] The proximity of a cloudlet to its associated mobile devices is the key to its value in hostile environments. It's easiest to appreciate this in the context of DoS attacks on wireless backhauls.

Reducing DoS Vulnerability

Consider DoS attacks through wireless jamming in military operations. *Range* and *directionality* of wireless transmissions are the primary levers of control available to a mobile device in trying to reduce its DoS attack surface. Physical layer mechanisms, such as frequency hopping and spread-spectrum transmissions, are also relevant. Choosing these parameters wisely can greatly increase the work factor needed for a successful DoS attack.

If the cloud is located near the mobile device (ideally one wireless hop away) and ultra-short-range wireless technology is used, then a very high work factor is needed for a successful DoS attack. In this case, only attacks from jamming sources that are physically close to the mobile device are threats. If an area larger than the jamming radius can be physically secured around the mobile device, jamming will no longer be a threat. Directional transmissions can further shrink this area.

By using only a single wireless hop, threats unique to multihop networks can also be avoided. In other words, placing the cloud close to its mobile device transforms the difficult and poorly understood problem of DoS attacks into the more tractable and well-understood problem of physical security. In the limit, the work factor for a successful DoS attack is increased to that of physical capture of the mobile device.

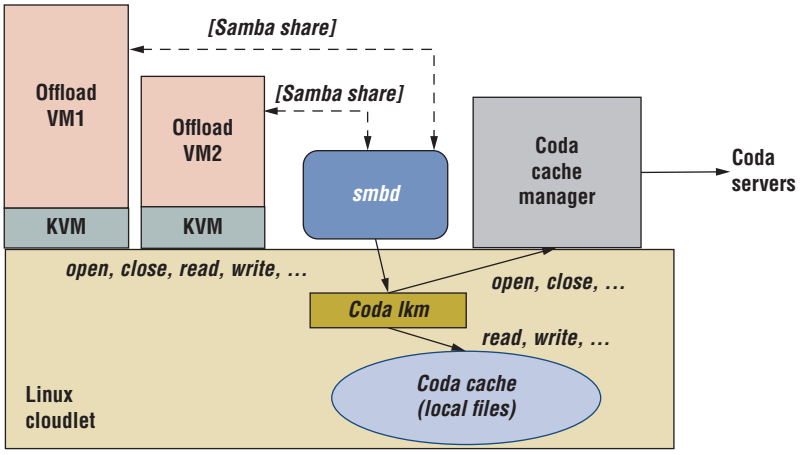


Figure 4. Virtual machine access to shared data. This offers strong file system consistency for offload operations.

Cloud proximity also reduces information leakage for traffic analysis. Inferring an imminent operation merely from recent changes in wireless traffic volumes and patterns (even without access to the data content) is a capability of long-standing military importance. Restricting the range of end-to-end communication denies distant snoopers access to that traffic information.

Achieving Resiliency

How can we leverage cloudlet proximity to improve resiliency? The approaches we describe here involve relatively small modifications to existing cloud-based software. So, the path forward doesn't have to be disruptive; it can build on a rich body of existing software.

Transforming a mobile application that offloads to the cloud is relatively simple, provided its back end is already encapsulated in a VM. We only have to ensure that the mechanism for selecting an offload site favors nearby cloudlets. A potential complication is ensuring that the selected cloudlet has a copy of the necessary VM image. Fortunately, dynamic VM synthesis has been shown to be a good solution to this problem, achieving just-in-time provisioning of a cloudlet in 10 to 15 seconds for typical VM image sizes.[13]

Synchronizing multi-user collaboration in a hostile environment is also simplified by using a cloudlet. By definition, a cloudlet is well-connected to its associated mobile devices. By running the synchronization software on a cloudlet (rather than the cloud), a team of mobile users associated with that cloudlet can easily collaborate—no software changes are necessary. This extends to cases where the team is spread across multiple cloudlets, provided those cloudlets are well-connected (as in Figure 3). Only when a team is spread across multiple cloudlets that might be partitioned from each other does this approach fail, and there might not be any feasible solution in this case. Fortunately, many team-oriented actions are likely to occur in situations where the entire team shares a single cloudlet (for example, a platoon or company in a military operation, with the cloudlet located in the team leader's vehicle).

The most difficult problem is cloud-sourced data access. The fundamentals of disconnected and weakly connected access to cloud-sourced data were described nearly two decades ago in the context of the Coda File System.[4,14] The essential steps are hoarding (prefetching of data into a cache), emulation (masking the absence of the cloud during failures), and reintegration (propagating updates to the cloud and resolving conflicts). On a cloudlet, there's a choice between maintaining a single cache in the host on behalf of all its VM instances or requiring each VM instance to maintain its own cache within its guest.

In most cases, a single cache per cloudlet will be preferable. This cache can be exported to each VM instance through a Samba share, as Figure 4 shows. This offers strong file system consistency for offload operations that, for example, use a collection of VMs on a many-core cloudlet to implement a MapReduce task.

A Samba export to each mobile device associated with the cloudlet is also feasible, assuming excellent Wi-Fi connectivity. This configuration ensures that the entire collection of mobile devices associated with a cloudlet and their offload VM instances see a strongly consistent shared file system. Only with respect to the cloud is consistency weakened. In that case, the classic Coda consistency guarantee applies: one-copy semantics at open-close granularity at all connected sites and eventual consistency at all currently inaccessible sites. Another possibility is for each mobile device to maintain its own data cache. This would lower Wi-Fi bandwidth demand at the cost of weaker consistency. A third possibility is to create a hybrid cache consistency protocol that has a peer-to-peer component (between a cloudlet and its associated mobile devices), and a client-server component (between the cloud and cloudlet).

Today, not all cloud-sourced data is stored in a distributed file system. Many specialized data repositories, such as Google Maps, YouTube, Flickr, GigaPan, and Netflix, have wide variance in attributes such as read-write ratio, data size, and

concurrency control. The Coda approach can be used as a template for making each of these resilient to cloud-cloudlet disconnections. A proxy of the repository is instantiated on each cloudlet. This effectively serves as a cache manager, hoarding data from the cloud in anticipation of disconnection.

The hints for hoarding can, in many cases, be inferred from the context. On a map server, for example, the geographical region around the cloudlet can be assumed to be of highest importance. Similarly, for a face recognition database, the faces of people who reside in the local region are likely to be most important. Other sources of context, such as social media (Facebook or Google+, for example), can also be used for hoarding hints. Mobile devices associated with the cloudlet direct their requests to the proxy rather than to the cloud. The update path (adding a new YouTube video, for example) will need to be specific to each type of repository. As in Coda, a log of updates awaiting replay to the cloud can be maintained by the proxy. Those updates are immediately visible to all the mobile devices associated with the cloudlet. A cloudlet and its associated mobile devices can thus form a self-contained world that's only loosely dependent on the cloud. In the terminology of Herb Simon,[15] the use of cloudlets transforms a cloud-based system into a *nearly decomposable system*.

Supporting Heterogeneity

As mentioned earlier, rescue operations in disaster recovery are often hindered by a lack of software interoperability across responding teams, each with its own set of mobile devices and applications. The need for rapid response and the lack of software expertise in the field imply that it's unwise to rely on near-perfect advance coordination or complex software setup. Self-configuring approaches based on a few simple building blocks are preferable.

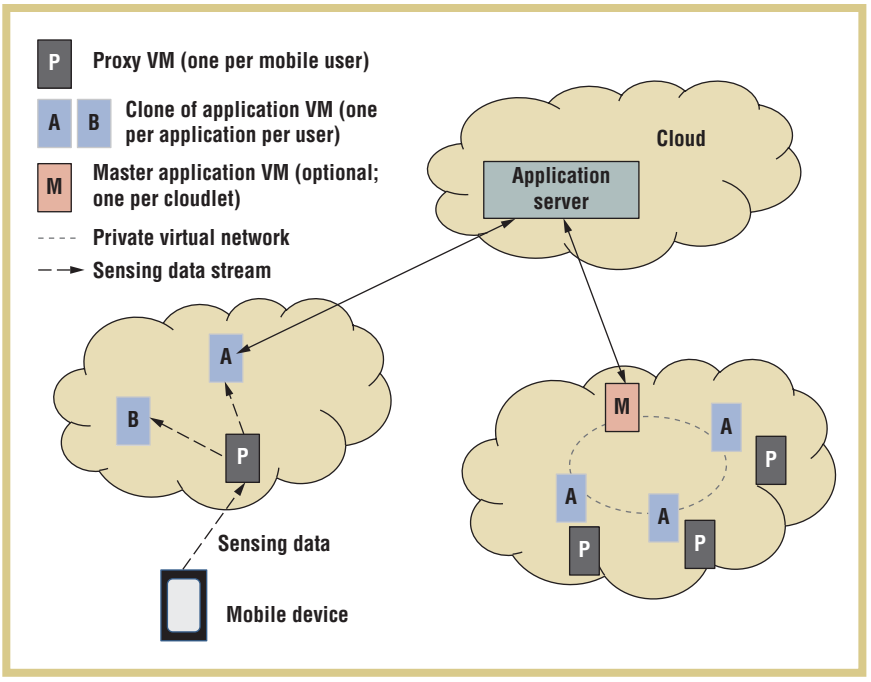


Figure 5. Proxy VMs in cloudlets.[16] The proxy VM mediates all interactions by the mobile device with other mobile devices or with cloud/cloudlet services.

Although heterogeneity is a difficult problem to solve in its full generality, VM-based cloudlets can serve as helpful infrastructure that simplifies the development of specific interoperability solutions. A VM cleanly encapsulates and separates a user-specific guest environment from the cloudlet-wide host environment. The interface between the host and guest environments is narrow, stable, and ubiquitous. The malleable software interfaces of operating systems, dynamically linked libraries, and applications are encapsulated in the guest environment and are thus precisely preserved and transported over time and space.

A specific path forward is suggested by Yu Xiao and her colleagues' recent work on the use of cloudlets to scale up crowd-sensing applications.[16] That work proposes representing each mobile device using a *proxy VM* on the device's associated cloudlet. As shown in Figure 5, the proxy VM mediates all interactions by the mobile device with other mobile devices or with cloud and cloudlet services. The proxy VM thus provides a level of indirection that can be leveraged in many ways to cope with specific aspects of heterogeneity.

Challenges

Realizing the potential of cloudlets in hostile environments will require overcoming many challenges. We sketch some of these below, recognizing that our discussion is necessarily brief and incomplete due to space limitations.

Perhaps the biggest open question is whether the cloudlet concept will gain sufficient traction to stimulate widespread deployment and investment. Fortunately, the trend is promising. Microdata centers, from companies such as Myoonet (www.myoonet.com/unique.html) and AOL,[17] are already available for repurposing as cloudlets. Because these appliances are designed for rapid deployment as private clouds (level 1 data centers) rather than as cloudlets (level 2 data centers), changes will be needed in their operational software environment.

IBM and Nokia Siemens Networks recently announced a collaboration to create a mobile edge computing platform that can run application software.[18] This is suggestive of industry alignment with the underlying concept of cloud proximity. A key concern is fragmentation of the marketplace due to the deployment of cloudlets with proprietary software interfaces. This can be avoided if a standardized software interface to cloudlets is widely embraced, starting from a base such as OpenStack (www.openstack.org).

The issue of trust in cloudlets will be a significant challenge. Today, a (level 1) data center is effectively a small fort, with careful attention paid to physical security of the perimeter. Hardware tampering within level 1 is assumed to be impossible. Mechanisms such as attestation based on a trusted platform module (TPM) are therefore not often used at this level. In contrast, a cloudlet has weak perimeter security, even if it's located in a locked closet or above the ceiling. Consequently, trust-enhancement measures, such as tamper-resistant and tamper-evident enclosures, remote surveillance, and TPM-based attestation, will be important. How to balance trust with ease of deployment remains an open question.

Cloudlet discovery and association pose new challenges in hostile environments. Attributes such as wireless jamming resistance, physical safety of location, and stable power must be factored into the cloudlet choice. Manual selection, using a mechanism similar to what's already in use today for choosing Wi-Fi networks based on their SSIDs, is one possibility. More sophisticated solutions could be modeled after service discovery mechanisms such as UPnP, Bluetooth Service Discovery, Avahi, and Jini.

Developing support for disconnected operation of diverse types of cloud-sourced data will need to be an important area of effort. The problem has been explored in depth for hierarchically structured distributed file systems. Leveraging that rich source of knowledge to other types of data will require significant innovation in many areas. These include mechanisms for hoarding cloud-sourced data in anticipation of disconnection, for emulating services when disconnected, and (for mutable data) for reintegration and conflict resolution of updates made while disconnected.

Today, in mobile platforms such as Android, applications don't interact directly with a file system interface even though their persistent data is stored in an underlying file system. It's therefore necessary to bridge the large semantic gap between application-level abstractions and the hoarding, emulation, and conflict-resolution capabilities of the underlying disconnectable file system. A toolkit or library approach to accomplish this is one possibility. An alternative is to leverage a Web-based model, such as HTML5, that's disconnection-aware.

We'll also need to realize the full potential of VM-based mechanisms in bridging heterogeneity. A proxy-based approach offers promise, but the details must be worked out for specific combinations of software environments that are likely to be used by collaborating parties. Developing toolkits to simplify this effort will be valuable.

In its infancy, mobile computing was characterized by devices that operated as standalone units, with no dependence on external resources. Over time, the scope of mobile computing has expanded to many applications that combine local sensing and user interaction with remote data access and compute-intensive processing. As dependence on external resources grows, so does vulnerability to their absence.

Here, we've identified a class of environments in which cloud access is frequently and unpredictably disrupted. Today, such hostile environments are relatively rare, appearing only during events such as military operations and disaster recovery. However, if cyberwarfare becomes more prevalent, the entire public Internet might have to be viewed as a hostile environment.

The central message of this article is that physical proximity is a precious attribute in matters of robustness and survivability. The much-heralded "death of distance" claimed for the Internet fails to recognize that wide-area communication, especially wireless communication, is easy to disrupt. We advocate a more conservative design strategy: cloudlets that are just one wireless hop away from their associated mobile devices serve as physically proximate representatives of the cloud. Such a tiered approach preserves the benefits of cloud-mobile convergence while improving survivability. P

ACKNOWLEDGMENTS

This material is based primarily on work funded and supported by the US Department of Defense under contract no. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. This material has been approved for public release and unlimited distribution (DM-0000276). Additional support was provided by the US National Science Foundation (NSF) under grant numbers CNS-0833882 and IIS-1065336, and by Intel, IBM, Google, and Bosch.

REFERENCES

1. S. Brenner, *Cyber Threats: The Emerging Fault Lines of the Nation State*, Oxford Univ. Press, 2009.

2. A. Savvas, "Firms Will Flee Cloud if Lessons from Siri and RIM Outages Not Learned," *CFO World*, Nov. 2011; www.computerworlduk.com/news/cloud-computing/3316907/firms-will-flee-cloud-if-lessons-from-siri-and-rim-outages-not-learned.

3. D. Kucera, "Amazon Apologizes for Christmas Eve Outage Affecting Netflix," *Bloomberg Businessweek*, 31 Dec. 2012; www.businessweek.com/

news/2012-12-31/amazon-apologizes-for-christmas-eve-disruption-affecting-netflix.

4. M. Satyanarayanan et al., "The Case for VM-based Cloudlets in Mobile Computing," *IEEE Pervasive Computing*, vol. 8, no. 4, 2009, pp. 14–23; doi: 10.1109/MPRV.2009.82.
5. J. Flinn. *Cyber Foraging: Bridging Mobile and Cloud Computing via Opportunistic Offload*, Morgan & Claypool, 2012.
6. J. Morris et al., "A Distributed Personal Computing Environment," *Comm. ACM*, vol. 29, no. 3, 1986, pp. 184–201.
7. M. Satyanarayanan et al., "The ITC Distributed File System: Principles and Design," *Proc. 10th ACM Symp. Operating System Principles*, ACM, 1985, pp. 35–50.
8. M.J. Zieniewicz et al., "The Evolution of Army Wearable Computers," *IEEE Pervasive Computing*, vol. 1, no. 4, 2002, pp. 30–40; doi: 10.1109/MPRV.2002.1158276.
9. "Information Warfare: Evolving Offensive and Defensive Information Warfare Strategies in Mobile Networks," white paper, Scalable Network Technologies, 2008.
10. J. Kong, X. Hong, and M. Gerla, "A New Set of Passive Routing Attacks in Mobile Ad Hoc Networks," *IEEE Military Communications Conf.* (Milcom 03), IEEE, 2003; doi: 10.1109/MILCOM.2003.1290214.
11. R.R. Rao, J. Eisenberg, and T. Schmitt, eds., *Improving Disaster Management: The Role of IT in Mitigation, Preparedness, Response, and Recovery*, National Academies Press, 2007.
12. J.J. Kistler and M. Satyanarayanan, "Disconnected Operation in the Coda File System," *ACM Trans. Computer Systems*, vol. 10, no. 1, 1992, pp. 3–25.
13. K. Ha et al., "Just-In-Time Provisioning for Cyber Foraging," *Proc. 11th Ann. Int'l Conf. Mobile Systems, Applications, and Service* (MobSys 13), ACM, 2013, pp. 153–166.
14. M. Satyanarayanan, "The Evolution of Coda," *ACM Trans. Computer Systems*, vol. 20, no. 2, 2002, pp. 85–124.
15. H.A. Simon, "The Architecture of Complexity," *Proc. Am. Philosophical Society*, vol. 106, no. 6, 1962, pp. 467–482.
16. Y. Xiao et al., "Lowering the Barriers to Large-Scale Mobile Crowdsensing," *Proc. 14th Workshop Mobile Computing Systems and Applications* (Hotmobile 13), ACM, 2013; doi: 10.1145/2444776.2444789.
17. R. Miller, "AOL Brings Micro Data Center Indoors, Adds Wheels," Data Center Knowledge, Aug. 2012; www.datacenterknowledge.com/archives/2012/08/13/aol-brings-micro-data-center-indoors-adds-wheels.
18. "IBM and Nokia Siemens Networks Developing Mobile Edge Computing Platform," *Cellular-News*, Mar. 2013; www.cellular-news.com/story/58971.php.

the AUTHORS

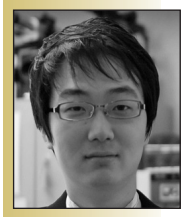
Mahadev Satyanarayanan is the Carnegie Group Professor of Computer Science at Carnegie Mellon University. He is an experimental computer scientist who has pioneered research in distributed systems, mobile computing, and pervasive computing. Satyanarayanan received his PhD in computer science from Carnegie Mellon University. He is a Fellow of the ACM and IEEE. He was the founding program chair of the HotMobile series of workshops, the founding editor in chief of *IEEE Pervasive Computing*, and the founding director of Intel Research Pittsburgh. Contact him at satya@cs.cmu.

Grace Lewis is a senior member of the technical staff in the Advanced Mobile Systems Initiative at the Software Engineering Institute. Her research and interests are in mobile computing, cloud computing, and service-oriented architecture. Grace received her master's degree in software engineering from Carnegie Mellon University, and she is currently a PhD candidate in computer science at VU University Amsterdam. She is an IEEE Senior Member and a member of the Executive Committee of IEEE Computer Society's Technical Council on Software Engineering. Contact her at glewis@sei.cmu.edu.

Edwin Morris is a senior member of the technical staff at the Software Engineering Institute. His current research focuses on the use of mobile computing devices and service-oriented computing in tactical environments, identity management approaches for situations where handheld devices are used, and end-user programming with handheld devices. Morris received his MS in computer science from Bowling Green State University. Contact him at ejm@sei.cmu.edu.

Soumya Simanta is a senior member of the technical staff at the Software Engineering Institute. His current areas of research are mobile computing in resource-constrained environments, and edge analytics. Soumya holds an MS in software engineering from Carnegie Mellon University. Contact him at ssimanta@sei.cmu.edu.

Jeff Boleng is a senior member of the technical staff at the Software Engineering Institute, working on the Advanced Mobile Systems team. His research interests include network protocols, operating systems, distributed computation, embedded systems, numerical analysis, scientific computing, parallel processing, and concurrency. Boleng received his PhD in mathematical and computing sciences from the Colorado School of Mines. He is a member of the ACM, IEEE, and the IEEE Computing Society. Contact him at jlboleng@sei.cmu.edu.



Kiryong Ha is a PhD student at Carnegie Mellon University, currently working on the convergence of mobile computing and cloud computing. His research interests include mobile computing, virtualization, and cloud computing. Ha received his MS in biosystems from KAIST. Contact him at krha@cs.cmu.edu.