3-2018

Nocloud: Experimenting with Network Disconnection by Design

Reza Rawassizadeh
University of Rochester

Timothy Pierson
Dartmouth College

Ronald Peterson
Dartmouth College

David Kotz
Dartmouth College

NoCloud: Exploring Network Disconnection through On-Device Data Analysis

Reza Rawassizadeh
University of Rochester

Timothy J. Pierson
Dartmouth College

Ronald Peterson
Dartmouth College

David Kotz
Dartmouth College

Application developers often advocate uploading data to the cloud for analysis or storage, primarily due to concerns about the limited computational capability of ubiquitous devices. Today, however, many such devices can still effectively operate and execute complex algorithms without reliance on the cloud. The authors recommend prioritizing on-device analysis over uploading the data to another host, and if on-device analysis is not possible, favoring local network services over a cloud service.

The paradigm of cloud computing has transformed the IT industry, enabling developers to use high-performance hardware and applications without raising a large amount of capital.[1] This type of architecture offers several other advantages, including significant reductions in hardware maintenance costs, scalability, and so forth.

To take advantage of the cloud, small devices such as mobile phones or smartwatches sometimes transfer data to the cloud for storage and processing, that is, they offload these functions. For instance, physical activity information derived from the accelerometers of wearables are often transferred to and stored in the cloud—as with, for example, Google Fit (www.google.com/fit) or Intel Context Sensing SDK (software.intel.com/en-us/context-sensing-sdk). Many virtual assistants, such as Apple's Siri or Amazon's Echo, transfer voice input to a vendor's cloud for analysis. The ability to offload complex tasks from devices with limited computation capabilities to virtually limitless processing capacity in the cloud sounds appealing, but it ignores two major issues: threats to privacy from organizational and government surveillance, and advances in hardware capabilities.

Published by the IEEE Computer Society
1536-1268/18/$33.00 ©2018 IEEE

Unlike our interactions with traditional personal computers, some computing devices, such as mobile devices and wearables, are deeply personal and can have presence in our most private spaces. This pervasive presence means that they can recognize behaviors that we do not intend to share with others. The severity of privacy and security risks of wearable, mobile, and Internet of Things (IoT) devices can be significant. For example, consider the October 2016 denial-of-service (DoS) attacks that used IoT devices in the US to overwhelm the DNS infrastructure (www.npr.org/2016/10/22/498954197/internet-outage-update-internet-of-things-hacking-attack-led-to-outage-of-popula). Any device that is connected to the Internet and uses cloud systems is vulnerable to exploitation.

The challenge of reducing use of the cloud for mobile applications has been recognized before, and some researchers have proposed mechanisms to substitute the cloud with local networks, including cloudlets.[2] Nevertheless, even local-area networks are prone to attack or abuse; attacks on Wi-Fi networks and their connected devices can be launched by proximate attackers. For applications that collect personal data we suggest that developers design their applications by reducing their application's reliability on the network as much as possible, and thus reduce their application's vulnerability.

According to conventional wisdom, computationally complex algorithms should be run in the cloud. However, sending data to the cloud could increase device energy use, the use of network bandwidth, and response time. We do not recommend avoiding the cloud or removing network connections in all scenarios—for example, it is not possible with social networking applications; rather, we recommend that developers avoid treating the cloud (or networks) as the default for hosting, managing, and maintaining users' data, or for all data processing. For instance, there are algorithms such as deep learning that are computationally complex and thus resource intensive. These algorithms provide high accuracy for some tasks such as image recognition. They, however, could not be hosted on traditional information-processing chips. In these scenarios, uploading data into a cloud is more cost-effective. In contrast, many ubiquitous and pervasive applications could work fine without any network connection or by using a hybrid approach that does not send all data into the cloud. There are models[3–4] to assist developers in calculating whether it is worthwhile to send the data to a remote host or not. We describe these later in the Related Work section.

On the other hand, hardware and devices are becoming ever more capable while decreasing in size and weight through miniaturization. Therefore, given the privacy risks and network and energy costs, we believe designers should default to a "local first" approach, that is, avoid using cloud services whenever possible, and furthermore, avoid the network altogether especially when handling sensitive data.

The decision to use cloud or local device depends on tradeoffs between communication and computation, application functions, network bandwidth, and energy costs. We suggest, if possible, applications stay disconnected from the network, especially the Internet, and perform analysis on-device and maintain data in personal storage. If on-device storage or processing is not possible, we recommend that developers favor a local network such as a cloudlet[2] over a general cloud service that is connected to the Internet. If there is an ultimate need for cloud uploading, we recommend developers consider making the offloading process a runtime decision, such as by using smart partitioning or dynamic offloading.[5]

We call our approach NoCloud and in this article discuss both its advantages and its challenges.

RELATED WORK

Cloud challenges have led to the introduction of concepts such as hybrid clouds,[6] cloudlets,[2,7] and fog computing.[8] Moreover, the cost of offloading computation to the cloud has been compared with processing data on a smartphone.[3,9–11]

Cloud Derivatives

The need for processing on-device and avoiding the cloud is well known for hostile settings such as military environments,[2] though the distinction between trustable and hostile environments is outside the capabilities of most end users. The hybrid cloud[6] is an architecture in which two clouds operate simultaneously: a local (private) cloud and a public cloud. The local cloud is similar to the external cloud but physically located near the target system. A cloudlet[7] is a private cloud server located close to the local system; an application triggers instantiation (or migration) of the necessary cloud service to a nearby cloudlet server.[10] It hosts the data and related processing. In that sense, a cloudlet is similar to traditional client/server architectures but supports multiple tenants via virtual machines.

Latency is an important issue in some IoT applications that require a near-real-time response. To improve cloud response time, Flavio Bonomi and his colleagues8 proposed the concept of fog computing, in which cloud nodes are geographically distributed and in closer proximity to the client device and thus lower the latency to near real time. A related approach is edge computing, in which vendors such as Cisco and Intel rely on local gateways to perform data preprocessing (in addition to routing data to the server).

Application Designs

Karthik Kumar and his colleagues[3] listed factors that should be considered while deciding whether to offload mobile computing to another host. These factors include bandwidth, server speeds, available memory, server loads, and the amounts of data exchanged between servers and mobile systems. Muhammad Habib ur Rehman and his coauthors[4] categorized three data-mining approaches for mobile phones: the on-board approach runs all processing on the device; the mobile approach processes data on a remote host; and collaborative approaches distribute the processing to an ad hoc network of connected mobile nodes in the same locality. Kyunghan Lee and his fellow researchers[11] quantified the impact of delaying data transfer in 3G networks, transferring data only when devices are connected to Wi-Fi.

So-called elastic mobile applications benefit from offloading and partitioning[12] the application process either statically or dynamically at runtime (for example, smart partitioning). In other words, they adaptively upload data to the cloud if there is a lack of resources on the local device; otherwise, the process will be done entirely locally. These applications usually upload intensive processes into the cloud such as natural language processing (NLP), and keep simple processes, such as user interface interaction, on the device.

Muhammad Shiraz and his coauthors[13] provided a taxonomy of mobile offloading approaches based on six elements: framework nature (for example, entire application migration or application partitioning), objective function (for example, energy savings and/or bandwidth utilization), migration granularity (for example, entire process or class level), migration pattern (for example, application proxy or binary code migration), migration support (system or application level), and partitioning approach (static or dynamic). They also reviewed mobile offloading approaches that mitigate data safety and users' privacy.

MOTIVATION

Due to increasing surveillance by government agencies and companies as well as miniaturization trends resulting in greater hardware processor power, we believe that developers could design applications to run more processes locally. From a technical perspective, four factors motivate the introduction of NoCloud: privacy and trust, energy efficiency, network reliability and outages, and response time.

Privacy and Trust

There are three major privacy threat actors. First, Edward Snowden[14] revealed that surveillance by government agencies is much more extensive than previously suspected. Second, large corpo-

rations such as Google (www.cnet.com/news/google-closes-3-2-billion-purchase-of-nest) and Apple (www.wsj.com/articles/apple-targets-augmented-virtual-reality-with-hiring-acquisition-1454107392) are acquiring pervasive technologies that can quantify detailed aspects of our life. These technologies provide useful applications but pose a threat to privacy. For instance, a smartwatch could record all voices in the owner's vicinity. Even when manufacturers or service providers try to protect customers' privacy, vulnerabilities in the device or back-end services could leave customer data open to attackers. Third, cybercriminals are increasingly hacking into cloud services, as demonstrated by the theft of compromising celebrity photos from Apple's iCloud service in August 2014 (www.wikipedia.org/wiki/ICloud_leaks_of_celebrity_photos).

Governmental and corporate violations of privacy are usually supported by legal institutions, and victims usually lack the resources to contest such violations. With their organizational superiority, such attacks can be more successful and wide-ranging compared to those conducted by cybercriminals. Technologically superior adversaries might be able to defeat even strong security approaches such as the separation of encrypted data (in one vendor) and key material (in another vendor). Governments have also been known to use their authority and legal tools to compel disclosure of data or keys from vendors. Therefore, we believe keeping data private could mitigate risks associated with these organizations. Although a government agency could compel end users to provide their physical device for analysis, the cost of approaching each user individually is significantly higher than getting the data from a centralized location and dealing with one organization instead of a group of users.

Researchers and developers have proposed new software infrastructure and hardware devices to challenge these threats. While these are promising efforts, as soon as the data gets outside the device it is vulnerable to wireless and cloud-based attack.[13] By storing and processing the data in a local network server that is managed by the device owner and disconnected from the Internet, the attack surface will be reduced. In other words, the attacker would need to be in the local vicinity of the network, which depends on the network's radio range. Nevertheless, one can argue that the physically owned device is prone to theft, loss, and damage. We agree there is a tradeoff between the security of these approaches, but a personal device is owned by the user and the user is the sole controller of the device.

Table 1 summarizes existing threats to data repositories based on the STRIDE model (www.owasp.org/index.php/Threat_Risk_Modeling#STRIDE). Except for the "elevation of privilege" threat, we believe other threats have less chance of successful execution by using the on-device or local-network-only approach. Put simply, a cloud attacker can be anywhere in the world and still be successful. If the device is networked only to a local server, the attacker has to be in close proximity to the network; if the device is not networked, the attacker must have physical access to the device. Nevertheless, there are examples of emerging physical device attacks as well. For instance, at the time of this writing there are reports of federal agents demanding those seeking entry into the US for their phone password to search or download content from the device. Therefore, we cannot argue that storing information on physical devices is the ultimate solution. Also, it is not possible to archive large amounts of media such as games or high-quality movies on wearable or mobile devices, and they are prone to loss or damage as well. A personal network-disconnected storage solution such as external hard disk might resolve the need for disconnected storage.

Energy Efficiency

Two trends are increasing the feasibility of running complex processes on-device in an energy-efficient manner and reducing the need for transferring data to another machine: hardware performance improvements and miniaturization, and advances in energy-efficient algorithms that can run complex tasks.

Hardware trends have, for decades, provided a steady increase in the capacity and capability of mobile, wearable, and other pervasive devices. For example, Kryder's law states that storage media are decreasing in size and increasing in capacity on a logarithmic scale. These ongoing trends suggest that ubiquitous devices either already have or soon will have enough capability to host their own data for the long term.

Table 1. Threats to data repositories, based on the STRIDE model.

Data storage and analysis endpoint	Spoofing identity	Tampering with data	Repudiation	Information disclosure	Denial of service	Elevation of privilege
Cloud	Highly possible due to universal web accessibility of cloud data—for example, fishing web addresses.	Highly possible; could be done by a remote adversary that has gained accessed to the cloud.	Possible; a member of a cloud hosting organization or a subcontractor who has access to users' data can misuse it.	Possible if an adversary can get access to cloud infrastructure.	Highly possible and similar to other network servers that are available on the Internet.	Rarely possible, since physically stealing infrastructure is practically impossible and such a deep cyberattack also requires significant technical superiority.
Local network	Possible; the adversary would need to be in radio range of the host to gain access to the host machine.	Possible; an adversary can perform signal jamming in near proximity to the device and inject fake data.	Possible for other adversaries who have direct access to the servers.	Possible if the adversary is in proximity to the network.	Rarely possible, unless the local network has a connection to the Internet, such as a hybrid cloud; nevertheless, possible via signal jamming.	Only possible by staying in close proximity to the device and its network.
On-device	Only possible if the adversary has access to the physical device (steals the device).	Only possible if the adversary has access to the physical device (steals the device).	Only possible if the adversary gains direct access to the device (steals or destroys the device)	Only possible if the adversary has access to the physical device (steals the device).	Not applicable	Highly possible due to risk of device theft; the environment where the device resides is usually not as secure as cloud servers.

Similar advances are applicable in computation; indeed, mobile devices now integrate specialized coprocessors for sensor-data processing, encryption, and machine learning. Examples include the NVIDIA Volta Chip (www.nvidia.com/en-us/data-center/volta-gpu-architecture), Huawei Kirin 970 (consumer.huawei.com/en/press/news/2017/ifa2017-kirin970), Apple A11

Bionic Neural Engine (en.wikipedia.org/wiki/Apple_A11), Intel Movidius vision processing unit (www.movidius.com/solutions/vision-processing-unit), and Qualcomm Snapdragon Neural Processing Engine (www.qualcomm.com/news/releases/2016/05/02/qualcomm-helps-make-your-mobile-devices-smarter-new-snapdragon-machine).

Meanwhile, developers are trying to reduce the computational complexity of important data-analysis algorithms. Consider the algorithm used to generate a smartwatch user's profile,[15] which is based on a resource-efficient frequent-itemset mining prediction algorithm;[16] such algorithms are usually too resource-intensive to run on small devices. Figure 1 demonstrates the differences between generating the profile on the device itself and transferring the data from the smartwatch to a smartphone for further analysis. On-device analysis is more efficient in both response time (Figure 1a) and energy use (Figure 1b), especially as the size of the data grows (number of days increases). Note that these graphs report the energy cost of the smartwatch only. In the Transfer & Prediction scenario, the smartwatch transfers the data to the phone and receives the result, but the phone's energy costs are not included, suggesting that total energy costs are even higher than shown. In this case, at least, it is far more efficient to process the data on-device than to send it to another device for processing.

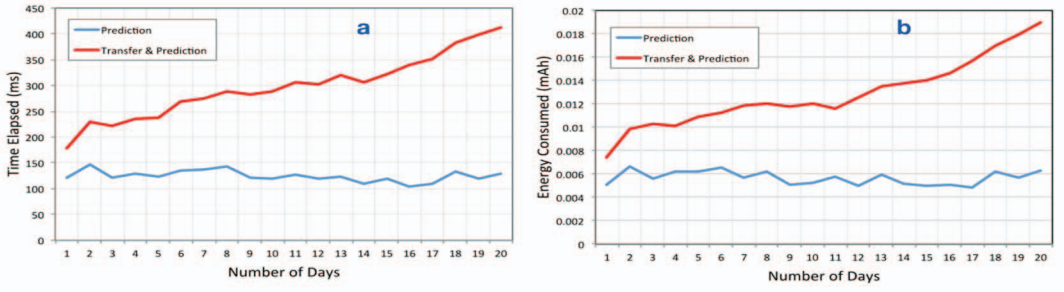

Figure 1. Latency (a) and energy use (b) of a smartwatch user profiling algorithm when run on the smartwatch alone (Prediction) and using a smartphone for data analysis (Transfer & Prediction).

Network Reliability and Outages

Many ubiquitous devices require Internet access to host and process device-collected data. Communications between geographically distributed nodes require Internet connectivity, and usually their webserver is hosted in a cloud. Communications between user devices, however, do not necessarily require Internet access (if they are in close physical proximity) and could be established in a local network.

Furthermore, the Internet is not available everywhere at all times, especially in underdeveloped regions, where broadband network penetration cannot keep pace with wearable market growth. For instance, the wearable market in Africa grew 89.9 percent in the first quarter of 2016.[17] Applications running on wearables could not proliferate in such regions unless they reduce their network need. In addition, medical devices that collect health data cannot rely on Internet availability in many rural areas, and natural disasters can temporarily disrupt Internet access even in well-served areas. Indeed, service outages can sometimes occur in enterprises with redundant connectivity options.[1] Regardless, any device or application that depends on the cloud or the Internet would become unusable during network unavailability periods. This dependence, and its effects on application availability, are another reason we recommend the NoCloud or local-first philosophy as a starting point for system design, wherever possible.

Response Time

Many ubiquitous applications involve real-time user interaction, demanding fast response times for any data-processing algorithms in the interaction loop. Sending the data to a network could

introduce unacceptable delay. Recall Figure 1a, which shows the response-time superiority for that particular algorithm by not transferring data to another device. It is notable that the latency problem of the cloud for IoT devices has been recognized and concepts such as fog computing proposed to resolve it.[7]

On the other hand, there are efforts to create lighter algorithms that can run on small devices, such as the user-profiling algorithm shown in Figure 1 or an NLP algorithm to query quantified-self data on the smartwatch.[18] Table 2 shows the response-time superiority of an NLP component running on a smartwatch compared with sending the data to a smartphone or the cloud. However, these queries do not require searching the Internet—they can be performed on-device because the information is collected by the device. The table reports the response time of a framework18 that evaluates parsing queries on-device with Google's SyntaxNet (github.com/tensorflow/models/tree/master/research/syntaxnet) and Apache's OpenNLP (opennlp.apache.org). SyntaxNet requires an external host, but OpenNLP has been implemented on smartphones.

Table 2. Response-time comparison between three libraries to parse quantified-self queries.

Method	Host	Response time (ms)
On-device parser	Smartwatch	594
SyntaxNet	Remote host (cloud)	3,639
OpenNLP	Smartphone	2,271

Although the quantitative tradeoffs will vary with the specific algorithm, the size and type of data, network bandwidth and latency, and the relative processing speed of the local and remote hosts, we believe it is not always faster to send the data to the cloud (or even to a nearby server) for processing.

Although we focused here on NLP, Shiraz and his colleagues[13] list several computational-intensive processes that are common among mobile devices including natural language translation, speech recognition, optical character recognition, image processing, online gaming, and video processing. Due to the changing dynamics of emerging pervasive devices such as companion robots, new algorithms might be required.

CHALLENGES

Of course, not every application can use on-device processing and data storage. Here we address five potential challenges of avoiding cloud services or favoring a local network server over the cloud: data integration, reliability and backup, information access, vendor financial incentives, and computational limitations.

Data Integration

One promised advantage of connected wearables or other ubiquitous devices is the opportunity to fuse the data with other information and thus make better inferences about human behavior than is possible with data from a single source. For instance, a wellness app could identify correlations between temperature and a user's physical activities (measured by a personal wearable device with on-premises sensor equipment). The key to the success of these types of algorithms

is the integration of data from several sensors. Although we envision many promising applications, we expect they will be difficult to deploy in the near future because of a lack of standard protocols available for communication between devices from different vendors. At this writing, there are more than 400 wearable devices available in the market (vandrico.com/wearables/list). Data integration will not succeed, we fear, until there are mandated or de facto standards defined, and service providers provide data in an open format. Furthermore, device/application vendors have many incentives *not* to enable their devices to interoperate with others: it is easier for them to develop integrated devices, apps, and portals in a vertical silo; a closed vendor-specific ecosystem produces "lock-in" that discourages customers from switching to competitors; and, finally, they may find value in secondary use of customer data. Unless or until there is a market benefit or legal mandate to interoperate, vendors will not be keen to support integration and standardization of their data.

There are some promising efforts to provide frameworks and standards for health data integration and device communication, such as Open mHealth (www.openmhealth.org), the Personal Connected Health Alliance (www.pchalliance.org), and IEEE 11073. They have not yet achieved widespread market adoption, however. Although data integration across devices and vendors and data sources is a compelling vision, the lack of interoperability and open protocols has thus far kept the market scattered across independent vertical silos. Without a need for data to "meet in the cloud," we again advocate that applications keep data close to their owners.

Reliability and Backup

The NoCloud approach encourages developing a local-first default architecture, in which all data is stored locally (on the mobile device, if possible) and processed there only. Mobile devices are prone to being lost or stolen, of course, requiring some data backup. A secure backup in the cloud could be one solution. Key management, however, is problematic: the encrypted backup can be stored in the cloud, but where does the consumer reliably and securely store the decryption key? What about a local backup solution, such as an external hard disk? If disconnected from the Internet it is not subject to network attacks, but it might still be vulnerable to malware if it is connected to a device that has been connected to the Internet before. Moreover, an external hard disk is also prone to theft and damage.

Both cloud and physical disk backup require the user to configure the system securely; configuration mistakes might expose data to attackers. It is hard to say whether it is easier to securely configure a local disk backup device or a remote cloud storage service; in either case, developers must ensure that secure configuration is simple for all users. On balance, we cannot recommend one solution over the other for all use cases.

Information Access

Usually, ubiquitous devices require an interface to present information they collect to the user. In many cases, devices either do not provide an interface or the interface is too small and limited to easily browse and search for information. Applications thus rely on complimentary devices such as a smartphone or web browser to display information to the user. For instance, physical activity data collected from a wrist-mounted wearable is commonly shown on a smartphone. The limited user interface of ubiquitous devices shows the importance of transferring data to another device, if only for display.

Many common devices transfer all device data to the cloud, then allow a web portal or mobile app to retrieve the cloud-stored data for immediate display to the user. We argue against this approach, for the reasons mentioned above; instead, the device should transfer data to the smartphone app only when display is required, and the smartphone could then discard the data once its display task is complete; the data never moves to the cloud. This method keeps the data secure inside the device most of the time, and exports a subset of the data only briefly and only when needed.

The transfer of data between the device and the display should be accomplished via a secure channel and confirmed by the device wearer. Methods have been developed for establishing secure, intended relationships between two devices—for example, LightTouch connects a smartwatch to an ambient display.19

Vendor Financial Incentives

In addition to hardware sales or software services, one of the most profitable assets of service providers is their access to user data. Indeed, there is a large market for data about individual users. For example, Google offers Gmail for free to users, but it makes profit on their data by using it to sell advertisements. With on-device data storage and analysis, such business models might not be sustainable due to the lack of access to customer data. This limitation could also affect quality of service, because customer data is often used internally by companies to improve their quality of service.

Without cloud storage, the device or service vendor does not have access to customer data, reducing profit opportunities; as a result, the devices or applications might be more expensive. Once again, there is a tradeoff; we believe users should have higher priority in making decisions about their data, and should be able to choose higher-privacy services even if they cost more. Some service providers mitigate this challenge by offering different levels of services, such as a discounted or free service that collects consumer data and more expensive services that do not provide consumer data to the vendor.

There is a general lack of privacy awareness among consumers.[20] Educating users about the privacy risks of pervasive devices could force vendors to change their policies and profit less from consumers' data.

Computational Limitations

One of the most important reasons for using the cloud is to benefit from access to a larger machine. Battery capacity (in wireless devices), computational ability, and radio signal strength are all affected by device size.[21] A smaller device usually means a weaker battery, less computational power, and shorter radio range. Nevertheless, we believe many of the requirements and functionalities of current ubiquitous systems should be implemented on-device and ubiquitous devices should move toward network independency. To demonstrate this need, consider deep-learning algorithms that have revolutionized machine-learning and data-mining applications in several domains, including image recognition. Hardware and algorithm improvements are bringing these new capabilities to mobile and wearable devices; as we noted above, GPUs and coprocessors are accelerating complex data-processing and machine-learning tasks on many such devices. We recommend that developers decide about the data process and storage host based on the algorithm's complexity. If the algorithm is resource intensive and the need for cloud uploading is inevitable, then developers can try dynamic offloading or smart partitioning rather than static offloading.

Another emerging trend is autonomous vehicles, including airborne drones that are disconnected from power sources. Drones can collect data, analyze it, and make inferences locally. Image recognition is a major task of drones, and image-recognition algorithms are usually computationally complex. Therefore, we anticipate future drones will be increasingly capable of conducting on-device data analysis. The popularity of these vehicles—and the wide range of practical applications—will drive the rapid improvement of both hardware and software for on-device data processing, thus reducing the need to transfer data to another machine for human intervention and decision making.

CONCLUSION

In this article, we recommend that system developers and researchers rethink using cloud services as a default architecture for mobile, wearable, and other pervasive computing applications and prioritize on-device analysis over uploading the data to another host. If on-device analysis is

not possible, then designers should favor a local network over the cloud. Our NoCloud proposal is motivated by four factors: privacy and trust, energy efficiency, network reliability and outages, and response time. Our approach has challenges with respect to data integration, reliability and backup, information access, vendor financial incentives, and computational limitations, but we outline how these can be overcome.

Mobile and wearable hardware are becoming increasingly capable as computational platforms, and we presented some examples to demonstrate that on-device processing can sometimes improve energy efficiency and response time relative to off-device processing. Moreover, we described industrial efforts to integrate computationally complex algorithms (such as deep learning) into the hardware, through coprocessors and GPUs. Although the specific tradeoffs are different for every application and network setting, NoCloud is a viable option for making decisions about architecture for ubiquitous systems. If there is no chance to host the process on the device, the application could send it to local network, and even if the local network is also incapable, developers can consider dynamic offloading methods and need not send data de facto to the cloud for further processing.

ACKNOWLEDGMENTS

We thank the editor in chief and the reviewers for their helpful comments. This research stems from a research program at the Institute for Security, Technology, and Society at Dartmouth College, supported by the National Science Foundation under award number CNS-1329686.

REFERENCES

1. M. Armbrust et al., "A View of Cloud Computing," *Comm. ACM*, vol. 53, no. 4, 2010, pp. 50–58.
2. M. Satyanarayanan et al., "The Case for VM-Based Cloudlets in Mobile Computing," *IEEE Pervasive Computing*, vol. 8, no. 4, 2009, pp. 14–23.
3. K. Kumar et al., "A Survey of Computation Offloading for Mobile Systems," *Mobile Networks and Applications*, vol. 18, no. 1, 2013, pp. 129–140.
4. M.H. Rehman et al., "Mining Personal Data Using Smartphones and Wearable Devices: A Survey," *Sensors*, vol. 15, no. 2, 2015, pp. 4430–4469.
5. B.G. Chun et al., "CloneCloud: Elastic Execution between Mobile Device and Cloud," *Proc. 6th Conf. Computer Systems* (EuroSys 11), 2011, pp. 301–314.
6. B. Sotomayor et al., "Virtual Infrastructure Management in Private and Hybrid Clouds," *IEEE Internet Computing*, vol. 13, no. 5, 2009, pp. 14–22.
7. M. Satyanarayanan et al., "The Role of Cloudlets in Hostile Environments," *IEEE Pervasive Computing*, vol. 12, no. 4, 2013, pp. 40–49.
8. F. Bonomi et al., "Fog Computing and its Role in the Internet of Things," *Proc. MCC Workshop Mobile Cloud Computing* (MCC 12), 2012, pp. 13–16.
9. E. Cuervo et al., "MAUI: Making Smartphones Last Longer with Code Offload," *Proc. 8th Int'l Conf. Mobile Systems, Applications, and Services* (MobiSys 10), 2010, pp. 49–62.
10. K. Ha et al., "Towards Wearable Cognitive Assistance," *Proc. 12th Ann. Int'l Conf. Mobile Systems, Applications, and Services* (MobiSys 14), 2014, pp. 68–81.
11. K. Lee et al., "Mobile Data Offloading: How Much Can WiFiDeliver?," *IEEE/ACM Trans. Networking*, vol. 21, no. 2, 2013, pp. 536–550.
12. D. Huang, P. Wang, and D. Niyato, "A Dynamic Offloading Algorithm for Mobile Computing," *IEEE Trans. Wireless Communications*, vol. 11, no. 6, 2012, pp. 1991–1995.
13. M. Shiraz et al., "A Review on Distributed Application Processing Frameworks in Smart Mobile Devices for Mobile Cloud Computing," *IEEE Comm. Surveys & Tutorials*, vol. 15, no. 3, 2013, pp. 1294–1313.

14. G. Greenwald, *No Place to Hide: Edward Snowden, the NSA, and the US Surveillance State*, Macmillan, 2014.
15. R. Rawassizadeh et al., "Energy-Efficient Integration of Continuous Context Sensing and Prediction into Smartwatches," *Sensors*, vol. 15, no. 9, 2015, pp. 22616–22645.
16. R. Rawassizadeh et al., "Scalable Daily Human Behavioral Pattern Mining from Multivariate Temporal Data," *IEEE Trans. Knowledge and Data Eng.*, vol. 28, no. 11, 2016, pp. 3098–3012.
17. C. Tredger, "Substantial Growth in Africa's Wearables Market," *IT Web Africa*, blog, 2016; www.itwebafrica.com/enterprise/505-africa/236475-substantial-growth-in-africas-wearables-market.
18. R. Rawassizadeh et al., "A Natural Language Query Interface for Searching Personal Information on Smartwatches," *Proc. 2017 IEEE Int'l Conf. Pervasive Computing and Communications Workshops* (PerCom Workshops 17), 2017, pp. 679–684.
19. X. Liang et al., "LightTouch: Securely Connecting Wearables to Ambient Displays with User Intent," *Proc. IEEE Int'l Conf. Computer Communications* (INFOCOM 17), 2017; doi.org/10.1109/INFOCOM.2017.8057210.
20. C.S. Lin, "Educating Students' Privacy Decision Making through Information Ethics Curriculum," *Creative Education*, vol. 7, no. 1, 2016, pp. 171–179.
21. R. Rawassizadeh, B.A. Price, and M. Petre, "Wearables: Has the Age of Smartwatches Finally Arrived?," *Comm. ACM*, vol. 58, no. 1, 2015, pp. 45–47.

ABOUT THE AUTHORS

Reza Rawassizadeh is a research associate in the Computer Science Department at the University of Rochester. His research interests include wearable and mobile technologies, machine learning, and data analytical algorithms for ubiquitous devices. Rawassizadeh received a PhD in computer science from University of Vienna. Contact him at rrawassizadeh@acm.org.

Timothy J. Pierson is a PhD student in the Computer Science Department at Dartmouth College. His research interests include privacy and security, especially as they relate to wireless sensor networks. Pierson received a BS in computer science from Michigan Tech and an MBA from Dartmouth College. Contact him at timothy.j.pierson.gr@dartmouth.edu.

Ronald Peterson is a senior research programmer in the Computer Science Department at Dartmouth College, and has held several software engineering positions in industry as well. His research interests include wireless sensor systems and wearable mobile health systems. Peterson received a BA in physics from Lawrence University. Contact him at rapjr@cs.dartmouth.edu.

David Kotz is the Champion International Professor in the Department of Computer Science at Dartmouth College. His research interests include security and privacy, pervasive computing for healthcare, and wireless networks. Kotz received a PhD in computer science from Duke University. He is an IEEE Fellow and a Senior Member of ACM. Contact him at kotz@cs.dartmouth.edu.