Survey of End-to-End Mobile Network Measurement Testbeds, Tools, and Services

Utkarsh Goel, Mike P. Wittie, Kimberly C. Claffy, and Andrew Le

Abstract—Mobile (cellular) networks enable innovation, but can also stifle it and lead to user frustration when network performance falls below expectations. As mobile networks become the predominant method of Internet access, developer, research, network operator, and regulatory communities have taken an increased interest in measuring end-to-end mobile network performance to, among other goals, minimize negative impact on application responsiveness. In this survey we examine current approaches to end-to-end mobile network performance measurement, diagnosis, and application prototyping. We compare available tools and their shortcomings with respect to the needs of researchers, developers, regulators, and the public. We intend for this survey to provide a comprehensive view of currently active efforts and some auspicious directions for future work in mobile network measurement and mobile application performance evaluation.

Index Terms—Mobile network, measurement, testbeds.

1 Introduction

Mobile (cellular) network applications deliver interactive services, generally supported by back-end logic deployed on cloud infrastructure. These applications support a wide breadth of functionality, such as live video, social gaming, communication services, and augmented reality [1]–[4]. Future services will increasingly leverage cloud-based datasets and processing power for innovative applications of live speech translation, real-time video analysis, or other computationally intensive tasks [5], [6]. As the frequency of interactions between mobile devices and back-end servers increases, application responsiveness will be increasingly tightly coupled with end-to-end network performance.

To innovate in the interactive mobile application space, developers deploy communication protocols with sophisticated data delivery techniques that support responsive communications under a range of network conditions [7]–[11]. These techniques are not always sufficient and developers are sometimes forced to redesign application functionality to mask poor network performance. However, these latter optimizations require detailed network performance data that is often not readily available, which results in challenges across the cellular ecosystem. For example, **developers** face the undesirable choice of evaluating performance of their mobile applications in limited private deployments that lack geographic diversity, or distributing their code to users without adequate testing [12], [13]. **Researchers** lack network performance data, or tools to acquire such data, in order to rapidly test hypotheses and focus on realistic network performance problems. **Network operators** need to monitor and troubleshoot end-to-end network performance without degrading base station throughput. Finally, **regulators** have a limited view of network performance, especially with respect to traffic shaping by network providers, impeding their ability to tackle performance challenges and roadblocks for sustained innovation in the mobile space [14], [15].

This paper provides a comparative analysis of currently available network measurement platforms for end-to-end mobile network measurement, monitoring, and experimentation. We further categorize measurement platforms as research testbeds for network experimentation, extensible distributed measurement tools, and services for widespread monitoring of networks performance. In the following sections describe the most salient features of each platform, and how some features differ across them. Table 1 compares the testbeds and tools in terms of their experimentation flexibility, device selection criteria, resource protection, and other features.

Based on our review of current measurement efforts, we observe that although existing approaches comprise only a patchwork of needed functionality, they already generate powerful insights to guide development, research, and regulatory actions. However, in spite of the relative maturity of several measurement platforms, daunting problems remain including support for wide-scale application prototyping and deployment, detection of traffic shaping, and long-term network performance monitoring. Most existing mobile measurement tools have been developed in isolation, and one motivation for this survey is to foster more concerted and cooperative efforts at standardization of measurement libraries,

- *U. Goel and M.P. Wittie are with the Computer Science Department, Montana State University, Bozeman, MT 59717.*
  E-mail: utkarsh.goel, mwittie@cs.montana.edu
- *KC Claffy is with UCSD/CAIDA, La Jolla, CA 92093*
  E-mail: kc@caida.com
- *A.Le is with Mintybit, Santa Barbara, CA 93111*
  E-mail: andrew@mintybit.com

	Network Testbeds						Network Tools							Network Services						
	Uncurated			Curated			Standalones					Libraries		Network Monitoring				Network Discovery & Diagnosis		
	MITATE	Seattle	PhantomNet	PhoneLab	SciWiNet	LiveLabs	FCC SpeedTest	WindRider	MySpeedTest	Mobitest	RILAnalyzer	Mobiperf	ALICE	Ookla SpeedTest	RadioOpt	OpenSignal	NetPerform	NDT	Netalyzr	PortoLan
Measurement Capabilities																				
Traffic shaping/DPI	✓	✓		✓				✓												✓
Active measurements	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓
Passive measurements				✓		✓		✓	✓		✓	✓	✓			✓	✓			
Measurement data publicly available	2			✓			2					✓	✓			✓		✓		
Custom packet content	✓	✓		✓						✓				1						
Peer-to-peer traffic	2	✓	✓																	
ICMP traceroutes	2			✓								✓	✓						3	✓
Programmable execution environment	4	✓	✓	✓		✓							✓						✓	
Access to mobile device sensor data	✓	2		✓	✓	✓		5	6		7	5,6	5,7	5	5,6, 7	5,6, 7	5,6, 7		5,7	✓
Experiments can be scheduled on specific clients	✓	✓	✓	✓		✓				✓			✓						✓	
IPv6 support	2	2										✓							✓	✓
Allow traffic on ports < 1024	3	3		✓																
Reports network problems												✓						✓	✓	
Supported mobile OS platform	8	8,9, 12	8	8	8	8,9	8,9	10	8	8,9, 11	8	8	8	8,9, 10,13	8,9, 10,11	8,9	8,9	8	8	8
Network coverage map																✓	✓			
Device Selection Criteria																				
Geographic location	✓	✓		✓	✓	✓										Q				
Device model	✓									✓		✓								
Device type (GSM/CDMA)																Q				
Battery charge level	✓	✓		✓												Q				
Carrier signal strength	✓			✓												Q				
Network carrier	✓												✓			Q				
Network type (Wi-Fi/Cellular)	✓			✓									✓			Q				
Time of day	✓	✓	✓	✓		✓				✓			✓							
Resource Usage Limit																				
Transmission rate		✓																		
Bandwidth cap	✓	✓			✓		✓		✓			✓	✓		✓					✓
Minimum battery charge	✓	2										✓								✓
Port restrictions		✓			✓															
Misc.																				
Measurement scheduling API	✓	✓	✓			✓						✓	✓							
Supports devices behind NAT/Wi-Fi	✓	✓		✓	✓		✓		✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓
Requires rooted phones			✓	✓						✓	✓									
Open to public	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
User incentive model	R	R,A		U		S,U	C	C	C	A		C,S	S	C	C,S	C,S	C,S	S	S	A
Experiments require IRB approval				✓		✓														
Open-source	✓	✓					✓	✓	✓	✓	✓	✓						✓		
Currently active	D	✓	D	✓	D	D	✓	✓	✓	✓	✓	✓	D	✓	✓	✓	✓	✓		✓
Records hardware specs	O				✓		✓	✓		✓		✓			✓	✓	✓			
Records hardware performance	O	O		O					✓	6					✓	✓				

Legends:

1 – measurements can be directed to specific Web servers.
2 – planned functionality.
3 – on rooted phones only.
4 – through multiple experiment rounds on the same device.
5 – GPS readings.
6 – battery readings.
7 – radio state.
8 – Android.
9 – iOS.
10 – Windows.
11 – Blackberry.
12 – Nokia.
13 – Amazon FireOS
A – user Altruism to support measurement capacity.
D – under Deployment.
O – Optionally.
C – user Curiosity to understand their own network performance.
Q – only at query time.
R – Reciprocal (tit-for-tat).
S – provides Service to clients other than measurement data.

TABLE 1
Experimentation flexibility matrix of end-to-end measurement testbeds, tools, and services.

privacy policies, and technology exchange [16]–[19].

The rest of this paper is organized as follows. Section 2 reviews goals of end-to-end mobile network measurement. Sections 3, 4, and 5 respectively discuss measurement testbeds, tools, and services for end-to-end mobile network measurement. Section 6 presents directions for future work and concluding thoughts.

2 Goals of End-to-End Mobile Network Measurement

The 2014 CAIDA workshop on Active Internet Measurements (AIMS 2014) brought together developers, researchers, network operators, and regulators interested in mobile (and wireless) network performance [18]. Although these communities share the goals of understanding and improving performance of current mobile networks, they focus on different metrics, and thus the tools they produce (Section 3) take different approaches.

2.1 Developers' View of Network Performance

Developers want to provide a responsive application experience to their users. Although much of the delay experienced by user requests is due to back-end processing and front-end rendering [10], as hardware and software processing speed improves, network latency becomes a dominant concern. Moreover, network latency does not necessarily benefit from advances in communication technology. Internet Service Providers (ISPs) typically engineer their networks to minimize forwarding costs, which is not always aligned with minimal end-to-end latency. Specifically, ISPs may direct traffic onto inexpensive but circuitous routes, which inflates hop counts and path latency [20]. ISPs may also configure cellular schedulers to delay transmissions until carrier channel conditions are favorable [21]. Additionally, mobile networks remain bandwidth-constrained, which motivates ISPs to deploy traffic shaping mechanisms on video streaming and P2P traffic to increase the usable bandwidth for all mobile users [22]. Traffic shaping can induce high latency that impedes the performance of dynamic content applications such as interactive Web, live video, and group communication and collaboration tools [23].

To cope with the complexity of mobile network performance dynamics, developers need to measure and incorporate mitigation strategies in their applications. Although not all mobile applications are equally affected by poor network performance, the responsiveness of network applications such as games, interactive video, and, to a lesser extent, in-car navigation and augmented reality requires low latency, stable bandwidth, or both [24]. To improve application responsiveness when latency is high, developers might redesign application communication protocols and message structures to pack more data in fewer round trips between mobile clients and back-end servers [25]. Developers might also strategically co-locate back-end servers in areas, or networks, where users tend to experience higher latencies [26]. To counteract the effects of low bandwidth, developers, might reduce the size and/or resolution of images and video, or reduce the frequency of application state updates by using techniques such as CloudFlare's Mirage [27] or Opera's Turbo [28].

To apply such performance adaptation techniques, developers need tools to study the performance of *their* application traffic in mobile networks to *their* application servers. Such *in situ* testing is useful during the application design process to reduce the risk of poor application performance at launch, especially when steep user base ramp-up is expected. For example, developers may want to evaluate a cloud server placement strategy and measure message delays across geographic areas to vali-

date whether application transactions meet their needs. Developers may also want to load-test their back-end infrastructure to ensure responsive service, irrespective of user location, server selection, and server load. Results from such analysis help mobile developers to design appropriate back-end deployment strategies. Network testbeds are useful for such early testing, because they save developers the trouble of writing testing code, or deploying dedicated back-end servers in multiple locations. Even during beta testing an application may not have a sufficiently large or distributed user base to generate statistically significant observations, and so community-maintained testbeds are a good alternative starting point.

Few testbeds support such realistic experimentation prior to application deployment. Most mobile network testbeds allow users to measure only upload and download speeds, ping latency, and traceroutes, but do not support prototyping of mobile application traffic, or detect traffic shaping in cellular networks. An alternative to public testbeds are paid services that evaluate application performance across multiple types of mobile devices. However, these services currently provide access only to stationary cellular devices, which limits measurement realism in terms of geographic and network diversity [12], [13].

Additionally, while there are published best practices for mobile developers [7], there are not many tools to track an application's communication performance throughout its lifetime. For example, developers may want to perform A/B testing to evaluate whether tweaks to communication protocols, or server endpoints, might improve performance. Such studies generally target certain users, networks, or time intervals, and thus require expressive test device selection criteria. Although A/B testing may be implemented in the application itself, third party application optimization libraries offer an easier, safer, and less disruptive starting point [29]–[31]. However, currently available A/B testing libraries focus primarily on testing application layout with respect to application adoption, user retention, and in-app revenue, but do not collect performance metrics needed to optimize application network performance.

Although server monitoring and reporting tools (e.g., [7]) enable logging and monitoring of application performance indicators such as request queue length at a server, they do not support end-to-end network measurement. Other analytics tools, for example the Google Analytics platform, provide performance measurement from a client perspective, but capture only the timing of the request-response cycle, and not response characteristics, e.g., size, compression, protocol [32]. Thus, developers need to measure and understand application performance in a realistic network environment before and after deployment, particularly as data needs and application requirements evolve.

2.2 Researchers' View of Network Performance

The research community has produced several testbeds that offer significant flexibility to execute a variety of network experiments [20], [33]–[45]. Yet, the availability of these testbeds and knowledge of how to use them often remains limited by practical barriers to collaboration across research groups. Researchers may need to set up their own infrastructure for data collection [46], obtain Institutional Review Board (IRB) approvals [34], or revive code that is no longer maintained [47], [48]. Even when maintainers of a given testbed help to set up experiments, communication rounds take time, especially when software modifications are needed. As a result, researchers often decide it is more expedient to develop new tools, even when it duplicates others' efforts and achieves only a small scale evaluation [34], [37].

Several organizations are working to lower the barrier to entry and promote concerted development of network measurement tools. For example, M-Lab maintains a repository of measurement tools, including MobiPerf, WindRider, and NDT (Mobile client), discussed in sections 4.2.1, 4.1.2, 5.2.1 respectively [49]. One of M-Lab's goals is for new tools to leverage existing code base, for example the Mobilizer library [41]. M-Lab also supports the development of common ethical guidelines for network measurement data collection [16]. However, the continued flow of proposals for new, independently deployed cellular tools (five in 2014 [37], [41], [44], [45], [50], eight in 2013 [20], [33]–[36], [42], [51], [52], three in 2012 [12], [40], [53], one in 2011 [38], one in 2010 [43], and two in 2009 [54], [55]) suggests that more needs to be done to improve collaboration among different research groups.

The research community has also worked to decrease the need for and the cost of redundant experimentation and created several repositories of wireless network measurement data [56]–[58]. While data repositories facilitate reproducibility of research results, they have their limitations. For example, to study current phenomena, such as changes in network traffic management policies expected after new FCC Net Neutrality regulations [59], researchers need new measurement data quickly, rather than waiting for a new dataset to be released after another group's publication. Additionally, data in repositories may be obfuscated, suitable for one experiment, but lacking in sufficient detail for another, or may be difficult to correlate when multiple datasets are collected at different times or under different conditions. For these reasons, live testbeds and measurement tools form a critical foundation of innovative research and education environments.

2.3 Network Operators' View of Network Performance

In addition to their operational monitoring of cellular network performance from base stations and other net-

work elements, network operators are also interested in end-to-end network measurement from the device's perspective, to provide responsive and reliable service at reasonable operating cost, including the cost of fielding customer support calls. Network operators also want to simplify and speed up the deployment of new access technologies and over-the-top services. A key element in these processes is the ability to troubleshoot network performance issues without affecting base station throughput.

However, industry insiders describe troubleshooting cellular networks as "an art with few scientific principles." To increase their insight into end-to-end network performance and network factors that may affect it, e.g, received signal strength, many network operators have deployed Carrier IQ on handsets in their networks [60], [61], and then faced customer backlash [62], [63] due to this application's approach (or lack thereof) to user privacy protection. Although network operators continue to use Carrier IQ, users continue to uninstall it on rooted phones [64], [65]. As a result network operators, like ATT, are looking for new methods to monitor and troubleshoot user network performance that can match the scale and efficiency of embedded end-host monitoring provided by Carrier IQ [66].

2.4 Regulators' View of Network Performance

Finally, regulators need monitoring tools to inform their understanding of availability, reliability, and performance of mobile networks over time. Constrained network performance and delayed upgrades to next generation technologies, e.g., 4G, have long been seen as stifling innovation in the US [67], [68]. Further, traffic shaping mechanisms and anti-competitive behavior by some network providers impede deployment of new services [15], [69]–[76]. Even developers of popular measurement tools struggle to create incentives for longitudinal and widespread measurement [52]. A few tools that have gained traction with users rely on user-initiated network tests, which limits measurement frequency and representativeness [38], [43].

2.5 Shared Challenges

Developers, researchers, network operators, and regulators face the same challenges in deploying end-to-end mobile measurement tools: incentivizing a statistically significant sample of users to install and execute the tool; protecting those users' resources from abuse; and preserving user privacy.

To motivate user participation, testbed designers have used schemes such as bundling testbed code with other functionality [46], offering free devices [34], press coverage [43], [52], or simply appealing to user altruism and curiosity [52]. These approaches result in either a narrowly focused user base or short-lived deployments, both of which limit testbed utility.

The second challenge is how to protect contributed testbed resources from abuse. Some peer-to-peer systems have used tit-for-tat mechanisms to ensure fair resource sharing [77], but mobile network measurement testbeds thus far rely on user altruism on the one hand and conscientiousness on the other [38]–[40], [52]. Scaling and sustaining measurement testbeds over the long term will require more rigorous resource protection methods in existing tools.

Finally, a testbed should isolate personally identifiable information from experimental data collected on a mobile device. Measurement tools discussed in this paper offer a range of solutions to maintain this separation. Google has supported the development of a proposed set of ethical guidelines for the design of mobile-based network measurement tools [16]. These guidelines have informed the design of some tools, specifically MITATE and Mobiperf, but the disparate legal frameworks for user privacy around the world make it difficult to create conformant tools for the global mobile Internet [18].

3 Network Testbeds

Mobile application developers need to know how well a network can deliver their application content. Custom network experiments that emulate communication protocols of their applications create performance profiles in different network settings to inform application design. End-to-end systems that support such functionality need to balance the flexibility of their feature set against potential abuse of contributed user resources and threats to user privacy. We divide systems according to how they resolve this conflict for new experiments from external researchers into uncurated and curated approaches.

3.1 Uncurated Network Testbeds

Uncurated network testbeds allow users immediate access upon registration. Users experiments and changes to these experiments do not need to go through an approval process. Although their open nature allows these platforms to scale, they are limited in the type of personal information they collect without going through an Institutional Review Board (IRB) approval process.

3.1.1 MITATE

Mobile Internet Testbed for Application Traffic Experimentation (MITATE), developed at Montana State University (MSU) in April 2013, enables experimentation with mobile application traffic in live mobile networks [20]. Experiments execute on user-volunteered devices that meet specified criteria, such as signal strength, geographic location, or network provider. Developers can use MITATE to evaluate the performance of mobile application communications under a wide range of conditions before their applications are deployed, or even fully developed. MITATE supports configurable active network measurements to detect network traffic shaping

by ISPs, and integration with other tools, for example CPLEX to explore protocol configuration tradeoffs through parameter search and optimization [78].

Functionality: MITATE supports active network measurements on mobile devices. MITATE experiments are configured through XML files that describe the content of experiment data transfers, transport layer protocols, network endpoints, and timing. An XML configuration also describes criteria that volunteered devices must meet to execute an experiment, such as network type (cellular or Wi-Fi), signal strength, geographic location, network carrier, minimum battery power, and device model. To ensure that experiments are defined correctly, MITATE servers validate new XML configuration files against an XML schema definition (XSD). Users interact with MITATE through an API that allows upload of XML configuration files and download of collected data.

Each mobile device polls a central MITATE server at MSU for new experiments whose criteria matches that device's capabilities. Devices download static traffic definitions that specify what traffic to exchange between the mobile device and back-end servers. MITATE mobile devices can interact with third party systems, for example DNS and CDN servers, through explicitly configured, well-formed request packets, and by recording reply content and delay. Although each MITATE experiment is a series of static transmissions, complex logic can be implemented across processing rounds, e.g., DNS lookups and ping transactions require two rounds. Such an experiment specifies a device ID as a criteria, which allows for the same device to issue DNS lookups in round one and subsequent pings in round two.

Data Collection: MITATE records the delay of each data transfer as well as metadata such as signal strength, accelerometer readings, and device location. This delay measurement allows calculation of 42 metrics, including uplink and downlink latency, throughput, jitter, and loss, as well as mobile sensor readings [79]. For example, an experiment estimates available bandwidth by dividing the size of a large transfer by its duration. MITATE experiments may also use a series of small transfers to estimate packet round trip time (RTT), loss, and jitter. At the start of an experiment, MITATE estimates the clock offsets between a device and each server endpoint, which allows separate measurement of uplink and downlink latency.

Collected data is available for download in the form of SQL insert statements to populate a local instance of a MySQL database for each user. MITATE allows users to download data only for their own experiments and those whose data is made public. Aggregate metrics, for example mean latency, are computed through queries to the local database instance. This design reduces the load on the MITATE database servers and allows users to run arbitrary queries over their experiment data.

Resource Incentives and Protection: MITATE is a collaborative framework built around incentives for user participation, inspired by BitTorrent's tit-for-tat mechanism [77]. MITATE users earn data *credit* by contributing their mobile resources. Users can then spend credit to run experiments on others' devices. Earned credit expires after 24 hours to prevent its accumulation and use for large experiments that might overwhelm available system-wide resources at any point in time.

MITATE's credit system encourages ongoing participation and protects contributed resources from abuse. Users can leverage MITATE resources in direct proportion to how much data they contribute to the system. MITATE does not rate-limit device transmissions (although users can set monthly data caps and battery limits on their devices), which permits realistic load-testing experiments. Although distributed denial of service (DDoS) attacks launched from multiple devices are technically possible in MITATE, they are destined to be short lived, because rapid transmissions from multiple devices will quickly deplete the malicious user's earned credit.

Privacy Protection: A significant challenge to expanding measurement systems on volunteered personal devices is the threat to user privacy. To limit the exposure of personally identifiable information, MITATE captures data only from active traffic experiments and does not monitor non-MITATE device traffic. Collected data is also indexed by virtual device IDs, rather than personally identifiable phone and International Mobile Equipment Identity (IMEI) numbers.

Remaining Challenges: MITATE is still in active development; project goals for the next couple of years include: deployment on M-Lab, support for peer-to-peer transmission between mobiles (important for IoT and gaming experimentation), and iOS device support.

3.1.2 Seattle

The Seattle testbed, originally developed in March 2009 at the University of Washington to support wired host experimentation, now also supports mobile application prototyping [80]. The design goal was to increase the diversity of testbed hardware to provide a more realistic prototyping environment than testbeds relying on dedicated hardware (e.g., PlanetLab, Emulab, or GENI [81]–[83]). Seattle runs on volunteered devices in last mile networks, and on institutional servers. As of 2015, Seattle includes about 800 mobile devices and over 10,000 nodes in total.

Functionality: Seattle experiments run on sandboxed virtual machines in a pared down implementation of Python called Repy. Seattle libraries support Repy functions such as data serialization, cryptography, and processing URLs, HTTP messages, and other protocols. Repy code is pushed to Seattle-registered through an API. Users can select devices by location and network

type (Wi-Fi or cellular) to which device is connected, but Seattle does not support selection by device travel speed, provider, or model. Seattle also supports P2P communication among devices.

Data Collection: Seattle does not collect network performance data by default. Instead users define their own metrics through experiments implemented in Repy. Seattle does not provide access to device sensors [84]. although sensor applications can make sensor data available to Repy programs through an API. The Sensibility testbed is an extension of Seattle, which allows Repy experiments to interact with mobile sensor data, but not to transmit or capture network traffic [85].

Resource Incentives and Protection: The Seattle incentive model is based on a tit-for-tat approach, where a user has access to ten volunteered devices for every device she registers with the system. While this policy makes sense in the wired setting, where devices are not generally restricted by monthly data caps, users who register wired hosts but experiment with others' mobile devices can deplete the mobile data cap. As a mitigating step, by default Seattle limits data transmissions to 10 Kbps, so even if the experiment fully uses that transmission rate, the owner can likely continue using their device. This limit prevents Seattle experiments from measuring available bandwidth and generating load-testing traffic – limitations not present in MITATE's credit-based model.

Privacy Protection: Seattle protects user privacy by allowing experiment code execution only in sandboxed virtual machines, which isolates experiment processes from each other and from non-Seattle processes.

Limitations: The authors of Seattle list several limitation of the current system, including inability for Seattle nodes to host services on ports below 1024, increase the transmission limit on donated resources, send ICMP traffic due to Repy restrictions, and put a limit on battery drain [33].

3.1.3 Emerging Systems

PhantomNet, being developed at University of Utah, is an emerging testbed based on a network of small-cell base stations connected through a software-defined network (SDN) backbone [45]. Users will be able to not only experiment with end-to-end services, but also modify backbone traffic forwarding for their experiments. PhantomNet devices will have dual-radio interfaces, which will allow integration with a reseller network, for example through SciWiNet. PhantomNet also leverages management tools from other systems, notably Emulab and Seattle. Currently, PhantomNet remains under development.

3.2 Curated Network Testbeds

Curated network testbeds vet network experiments prior to deployment. In particular, vetting involves passive monitoring experiments that collect privacy sensitive data, such as users' traffic, or location history and may need to go through an IRB review. Other experiments may require changes to the testbed itself and need to be approved by the testbed's developer team [86].

3.2.1 PhoneLab

PhoneLab is a programmable smartphone testbed, developed at the University at Buffalo in November 2013, to support flexible experimentation intended to emulate application deployment scenarios [34], [87]. PhoneLab experiments are implemented as mobile applications pushed to rooted Android smartphones given to student volunteers at the University at Buffalo. PhoneLab's model supports long-term, passive experiments that can record network transitions, battery drain, and use of other applications on the device.

Functionality: PhoneLab experiments are pushed to participants either via the Google Play Store, or separate as over-the-air updates. PhoneLab can benchmark third-party mobile applications without modifications to their code, which may be required in other testbeds. PhoneLab mobile applications can run experiments in the background or interactively. PhoneLab also supports experiments at the OS level, with modifications to the Android runtime system. Platform experiments are vetted by the PhoneLab development team and go through pre-deployment testing. Researchers submit experiments as XML configuration files that specify background experiments to start or stop, log tags to collect, and where to upload collected data. The PhoneLab Conductor fetches configuration files from PhoneLab servers and pushes them to testbed devices.

Data Collection: PhoneLab data collection relies on the Android logging interface, which gives experiments access to device operational data (such as phone status, battery level, etc.), as well as custom application log data. All log data is uploaded to the central server when a device is charging. When their experiment completes, users receive an archive of data that matches experiment tags from all devices that participated in their experiment.

Resource Incentives and Protection: Unlike MITATE and Seattle, which rely on volunteered devices, PhoneLab provides phones with discounted data plans to its participants. In spite of this incentive scheme, the PhoneLab team has faced significant participant attrition, with only 43 of 191 volunteers continuing after the first year [34]. PhoneLab limits the number of simultaneously active of experiments on each device to balance device utilization against interference between experiments.

Privacy Protection: To protect user privacy, experiments

submitted to PhoneLab need IRB approval or exemption. PhoneLab participants choose to participate in a particular experiment after reviewing what information will be collected. Participants can opt-out of an experiment at any time.

Limitations: PhoneLab's use of data plan subsidy potentially limits the scalability of the testbed. Also if phones are not replaced frequently, testbed hardware will eventually lag behind phone models used by the general public. Finally, PhoneLab code is not publicly available, which precludes the possibility of private deployments [88].

3.2.2 SciWiNet

Science Wireless Network (SciWiNet), being developed at Clemson University, is a NSF-funded re-seller of network infrastructure, based on Mobile Virtual Network Operator (MVNO) model, which provides the research community with a service on Sprint's cellular network infrastructure (and T-Mobile's infrastructure by late 2014) [44]. SciWiNet supports experimentation over 3G and 4G cellular networks, but without support for SMS, MMS, or voice services. SciWiNet provides additional infrastructure to the research community in the form of a shared pool of wireless devices (smartphones and USB LTE dongles), a common set of Android applications (WiFi hotspot, VPN tunnels, performance monitoring programs), and a set of wireless network services (VPN tunnel termination, secure database backend, performance monitor servers and backend).

Deployment: The SciWiNet project has two proposed project phases and is in phase-I as of September 2014. In phase-I, the project aims to determine the potential user community for SciWiNet infrastructure and investigate capabilities that it should support. In phase-II, the project will develop, deploy and operate the functional SciWiNet network infrastructure based on what was learned in phase-I.

Device support: Since SciWiNet uses Sprint's cellular network as its back-end cellular infrastructure, Sprint maintains a whitelist of mobile devices that are authorized to access SciWiNet's network and therefore eliminates the need to install a SIM card in every mobile device. Although SciWiNet records device MAC address, it does not make the device MAC publicly available. SciWiNet maintains a list of popular devices and blacklisted devices. iOS devices are excluded because they do not support re-seller networks [89]. SciWiNet helps researchers access testbed resources by providing them with 1-2 mobile devices and a prepaid data plan for a limited time, typically six months. Alternatively researchers can access SciWiNet from their own devices and SciWiNet covers part of the data usage costs.

Data Collection: SciWiNet Android app collects the following network measurements over cellular and Wi-Fi networks: throughput for TCP and UDP traffic flows, packet loss, and ping latency. It can also detect location-based services such as base station identity, location, and wireless signal strength.

Resource Incentives and Protection: Users can login to their account to check their data usage, or data contributed by others to their experiments. Data usage is limited by a leaky bucket rate limiter, where a user receives a number of tokens, which he can share among multiple devices. Once the data rate is exceeded, the device is temporarily restricted from accessing the SciWiNet network.

Remaining Challenges: As of September 2014, it is unclear how SciWiNet will provide access to its devices and network resources to the research and developer community. One possibility is to offer incentives for user participation by providing free or discounted device access.

3.2.3 LiveLabs

LiveLabs, designed at Singapore Management University in February 2014, is a mobile testbed intended to evaluate location-based services, such as commercial promotions to shopping mall customers [37]. LiveLabs has been tested on the campus of the Singapore Management University (SMU) and is currently being deployed at a large shopping mall near SMU campus, Singapore Changi International Airport terminal, and on the Sentosa resort island. The testbed is available to the three partnering venue operators, but not the general public.

Functionality: To facilitate evaluation of location-based services, LiveLabs supports device location discovery in indoor settings as well as characterization of user behavior. LiveLabs is designed for continual operation, thus the design has focused on low energy usage, for example by allowing multiple experiments to concurrently use sensor readings such as GPS, or WiFi signal strength. Researchers and participating companies use LiveLabs to evaluate location-based applications, for example real-time promotions to users at a shopping mall. LiveLabs is available for Android and iOS systems.

Data Collection: Unlike other testbeds discussed in this section, LiveLabs does not collect network performance metrics, but instead focuses on discovering user behavior, by recording device ID and a variety of sensor readings. The LiveLabs backend then supports higher level functions to detect and record user behavior, such as history of movement, group size, user physical queue length, and activities such as standing, walking, or sitting. LiveLabs also records information about participating users, such as their nationality.

Resource Incentives and Protection: LiveLabs has three

mechanisms for garnering user participation: rebates on users' monthly data bills; context-based apps that offer rebates on specific commercial services in deployment locations [90]; and a "lucky draws" lottery, though details of frequency and prizes are not specified [37].

Privacy Protection: Data collected by LiveLabs has the potential to disclose private user information, such as location, shopping patterns, and nationality. As such, experiments launched on LiveLabs go through SMU's IRB approval process [91]. Users are also asked to opt-in to data collection on their devices.

Limitations: LiveLabs is not designed for mobile network measurement (does not collect network metrics) and so it offers functionality distinct from MITATE, Seattle, and PhoneLab. At the same time, LiveLabs supports experimentation with new services in the mobile environment similarly to PhoneLab and has attracted participation of 30,000 users through its incentive model and business partnerships.

4 Measurement Tools

Mobile network performance characterization requires wide scale and ongoing measurement from a variety of devices across different networks and locations. Tools in this space, developed by industry, research, and regulatory communities, differ in how they obtain network metrics and how they select devices for measurement. Although network measurement tools presented in this section are not testbeds, in that they only support a fixed set of experiments, these tools do support long-term and wide-scale network monitoring, which offers important insights to developers, researchers, and regulators.

4.1 Standalone Measurement Tools

Standalone measurement tools are ready-to-deploy solutions with pre-defined network measurement functionality. The open-source nature of these tools allows other to modify them, although many of the tools offer measurement customization options. Data collected by these tools is generally, though not always, publicly available.

4.1.1 FCC Speed Test

The FCC Speed Test app, released in November 2013, was designed to provide insight to regulators and the public on the performance of mobile networks across the United States [52]. Developed in collaboration with SamKnows and major wireless service providers, the free application is available on Google Play Store for Android smartphones [92]. An iOS version of the application is also slated for release, though limitations of the iOS API prevent collection of some metadata that is collected by the Android version [14], [93].

Functionality: At the start of a measurement, the FCC Speed Test app pings available measurement servers to identify the one with lowest round trip time (RTT) to the mobile device. The selected server then sends a list of measurement instructions to the mobile device. If the mobile device is currently using less than 64 Kbps of bandwidth for other tasks, it starts the measurements, otherwise the device postpones measurement until its bandwidth usage drops.

The FCC Speed Test app supports active traffic measurements over four types of connections: single connection HTTP GET and POST, as well as multi-connection GET and POST. Multi-connection transfers test multithreaded download performance over three parallel downloads of 256KB data chunks. To measure packet loss and RTT, the FCC Speed Test app exchanges a series of UDP packets with the nearby server. Following a measurement, the mobile device uploads measurement data and associated metadata to an FCC server.

Data Collection: The FCC Speed Test app reports upload and download rates, packet loss, and RTTs based on HTTP and UDP transfers. Packet loss on a path is inferred based on failure to receive a UDP packet on that path within three seconds. The app records the number of packets sent each hour, the average RTT, total packet loss for performed tests, and throughput in 5-second intervals [94]. The app also collects device-related as well as network metadata, including signal strength reported by the device, connection type (3G/4G/Wi-Fi), location and ID of cell towers, GPS location, device model, OS version, network country code, SIM's operator ID, SIM's country code, network carrier, phone type (GSM/CDMA), and the device's roaming status.

Resource Incentives and Protection: To build nationwide measurement capacity the FCC Speed Test app relies on user curiosity about their network performance. Instrumental to the app's popularity and success was a press campaign [93], [95]–[98], which was followed by application installation and measurements from more than 50,000 devices in about 1.5 years. These numbers have declined over the life of the system, so the effectiveness of a publicity-driven approach to support long-term network monitoring remains to be seen.

Privacy Protection: The FCC app collects measurement data on the mobile device in the application sandbox, as opposed to through the standard Android logging interface, so data is not visible to other applications. The collected data are uploaded to FCC servers over encrypted connections. Once the data are uploaded, or become stale, they are automatically deleted from the application's sandbox storage. The FCC Speed Test app does not collect personally identifiable information, such as phone number or IMEI [99].

Limitations: The FCC Speed Test app executes only experiments configured by the FCC, i.e., it does not support

custom network measurement. As of October 2014, the configured tests do not detect traffic shaping in mobile networks, which is of increasing interest to regulators and the general public [15], [69]–[76]. With respect to resources used on the device, the FCC application runs at startup and prevents the phone from sleeping, which can drain the phone battery.

4.1.2 WindRider

Content-based traffic discrimination has recently been considered a threat to mobile application performance [15], [69]–[76]. WindRider, a measurement tool developed in 2009 at Northwestern University, detects application and service-based traffic discrimination by mobile ISPs [39].

Functionality: WindRider supports active and passive measurement of traffic shaping [54]. Active measurements exchange traffic between a user's mobile devices and a randomly chosen M-Lab server. The mobile device initiates a series of uploads and downloads and records their observed performance. To detect port-based traffic shaping, WindRider compares delay of identical transfers to different ports on M-Lab servers. Passive measurements record packet latency to well-known web servers during normal user browsing activity. To detect content-based traffic shaping, WindRider compares the observed packet delay to that reported by other devices in different carrier networks and locations to the same destinations. Active measurement results are stored on M-Lab servers, while passive measurement data, collected with user permission, are stored on WindRider servers.

Data Collection: The WindRider mobile application collects experiment-related data such as connection start time, connection establishment time, connection finish time, and number of inbound and outbound bytes [39]. WindRider also collects metadata such as device IMEI, device location (as ZIP code), network carrier, and browsing history. WindRider also collects device hardware performance metrics that can help interpret observed traffic delays, such as CPU execution time, virtual memory size, page faults per minute, and other metrics as permitted by the OS API.

Resource Incentives and Protection: WindRider relies on user curiosity for its network measurements.

Privacy Protection: WindRider optionally collects device IMEI, which can be linked with a user's browsing history. To protect user privacy, users can choose whether to make this information available to the application.

Limitations: Although WindRider supports detection of traffic shaping in mobile networks, it has two significant limitations. First, the measurement traffic is sent only to M-Lab servers, but developers may want to investigate traffic shaping on other paths. Second, WindRider only detects content-based traffic shaping as discrimination based on traffic sources, i.e., well-known Web servers, rather than type of traffic, for example BitTorrent.

4.1.3 MySpeedTest

The MySpeedTest mobile application, launched in June 2012 by Georgia Tech, measures network performance of mobile devices with the goal of observing and explaining patterns of user behavior in mobile ISPs to application developers [100], [101]. Such analysis may allow developers and service providers to tune application performance [102]. The MySpeedTest mobile application is available on Google Play and has more than 900 active users from 115 different countries, as of February 2013 [102]. As of April 2013, MySpeedTest is in the process of sharing a subset of their data with Google's M-Lab to help researchers benefit from data collected by each others' experiments [100].

Functionality: MySpeedTest performs passive and active measurements. Passively, MySpeedTest records the total number of bytes sent and received by each active application since the device booted. Information such as package name, bytes transmitted and received, application status (active vs. background) helps users know which applications consume the most data and power, and which applications may affect performance of other applications on the device.

Active measurements include a recurring test to measure TCP uplink and downlink throughput, inter-packet delay, and packet loss. MySpeedTest also measures network latency with 40 parallel ICMP pings to five servers in the U.S. and Europe. These tests store the minimum, average, and maximum latency to each of the five servers. The collected data help researchers and developers understand the performance of paths to potential application servers [102].

TCP-based experiments can reduce the bandwidth available to other applications on the device, so MySpeedTest performs TCP-based experiments only on user request, in a single thread for about 20 seconds, and using the maximum-sized packets that will not be fragmented. MySpeedTest also gauges streaming data quality by measuring packet loss and jitter of UDP traffic flows. MySpeedTest servers generates a stream of 64-byte UDP packets, transmission at Poisson-sampled intervals, with timestamps and sequence numbers in the payload. The server sends 500 packets with a data rate less than 1 Kbps to avoid congestion. The client calculates packet loss and jitter from every 10 packets received. The client compiles all data collected on mobile device into the JSON format and sends it to the server for storage.

Data Collection: The MySpeedTest mobile application collects experiment-related data such as TCP upload and download throughput, ping latency, UDP jitter, UDP

packet loss, and time to acquire a dedicated channel for data transmission [100]. MySpeedTest also collects device level data, such as cellular service provider, Android version, device manufacturer, connection type, radio firmware, hashed phone number, hashed IMEI, software version, SIM card state and serial number, latitude and longitude of base station, network operator ID, CDMA system ID, CDMA network ID, Wi-Fi signal strength, battery technology, status of battery charging, battery health, battery voltage, battery temperature, and device location.

Resource Incentives and Protection: Similar to the FCC Speed Test app, MySpeedTest relies on user curiosity about their network performance. MySpeedTest allows users to limit contribution of resources through a monthly data cap. To protect battery resources, MySpeedTest postpones experiments until the battery is above 5% and the device is attached to a network.

Privacy Protection: MySpeedTest collects personally identifiable information (phone number, IMEI, device location), which may expose private information, such as a user's location when a measurement occurred.

Limitations: Similar to MobiPerf and WindRider, MySpeedTest provides its users a limited network measurement capability between mobile devices and servers, as opposed to testbeds discussed in Section 3. MySpeedTest does not support transmission of custom traffic, such as tools to detect traffic-shaping based on content or port.

4.1.4 Akamai Mobitest

Akamai's Mobitest application and Web service, released in March 2012 by Akamai Technologies, measures the performance of mobile Web sites [12]. The application uses the WebPageTest framework and is available for Android, iOS, Blackberry based smartphones, tablets and simulators [103].

Functionality: Mobitest platform relies on user participation to install Mobitest software on their mobile devices. Each Mobitest installation on a device acts as an agent to the WebPageTest framework, where such device executes experiments requested by other users through the Mobitest Web service [104]. To measure the page load time on a mobile device, a user enters a URL through the Akamai Mobitest Web interface and selects the mobile device hardware that will perform the download [12]. Mobile devices running Mobitest periodically poll WebPageTest servers to obtain pending URL download requests entered by Mobitest users. Each requested URL is then accessed from the default browser on each device over the Wi-Fi, or cellular network, depending on how the device is connected at the time.

Data Collection: Akamai Mobitest collects the total time to load a Web page, individual request headers, average Web page size, as well as screen shots of the loaded page and optionally video of the loading page [104]. The tool produces waterfall charts of requests and delays, and an HTTP archive (HAR) file [105], [106]. The collected data helps researchers and developers gain insight into the responsiveness of Web servers and browser rendering of different site implementations [107]. Mobitest allows users to reuse previously collected measurements by linking them to user accounts on Akamai Mobitest's site.

Resource Incentives and Protection and Privacy: The Akamai Mobitest app allows application developers to set the frequency at which pending experiments are downloaded from WebPageTest servers to be executed on their mobile devices. Additionally, Akamai Mobitest allows users to control device resource utilization through a number of configuration options. Specifically, users can set whether the app should poll for new experiments after restart, whether to restart the app after every experiment, whether to capture network traffic, and the frequency at which screenshots for loading pages are taken [108].

Limitations: Akamai Mobitest evaluates the webpage load time on mobile devices, but does not allow more general experiments with non-browser-based application traffic, including how to characterize traffic shaping of non-Web traffic. The WebPageTest framework requires rooted phones, which limits the tool's applicability outside of dedicated test farms.

4.1.5 RILAnalyzer

RILAnalyzer, developed by the University of Cambridge and Telefonica in October 2013, is a client-side tool for monitoring of the mobile network control plane as well as the data plane [109], [110]. The application is available for rooted Android devices with Intel/Infineon XGold chipsets, which include the popular Samsung Galaxy S2/S3, Note 2, and Nexus devices.

Functionality: RILAnalyzer's focus is on discovering the promotions and demotions between the Radio Resource Control (RRC) states `IDLE` (no connection), `CELL_DCH` (dedicated communication channel), `CELL_FACH` (shared communication channel), and `CELL_PCH` (shared paging channel). Transitions between these states are triggered by control messages from the Radio Network Controller (RNC), which may themselves become a communication bottleneck [109]. As mobile devices consume different levels of energy in each of the RRC states, the devices themselves may use *Fast Dormancy* to reduce *tail-energy* and demote to lower energy states faster than through vendor and operator dependent timeouts [111].

RILAnalyzer implements a background tool that polls the device Radio Interface Layer (RIL) Daemon every second for the current RRC state. RILAnalyzer then

obtains data plane network and transport headers using NetworkLog [112] to identify applications active during each RRC state.

Data Collection: RILAnalyzer collects RRC states at one second intervals, headers and timestamps of outgoing TCP and UDP packets from NetworkLog as reported by the Linux kernel.

Resource and Privacy Protection: RILAnalyzer is intended for small scale studies on dedicated devices, or devices operated by expert users [109]. As such the tool's design has not made provisions to attract users with incentives, or to allow them to set limits on resource usage.

Limitations: RILAnalyzer is restricted to rooted phones on the Intel/Infineon XGold chipset. Although the authors of RILAnalyzer intend the tool for small scale studies, the specificity of hardware and overhead of reverse engineering RIL Daemon `OemCommands` commands does preclude large scale studies on diverse mobile hardware. RILAnalizer also puts a noticeable load on the CPU ($\sim 10\%$), memory ($< 42\%$), and storage (with packet logs), which may limit the willingness of volunteers to run the tool on their phones.

4.2 Libraries for Mobile Network Measurement

Libraries for mobile network measurement may be embedded in other applications to add network measurement functionality. This approach is potentially easier to adopt by Developers than extending open-source code of a standalone measurement tool. As in the case of Mobilizer, a library may also form a basis of a measurement tool, i.e. the current version of MobiPerf.

4.2.1 MobiPerf

The MobiPerf mobile application was developed as a collaboration of University of Michigan, Northeastern University, University of Washington, and Google's M-Lab to measure network performance and diagnose problems with application content delivery on mobile devices [38]. To allow the community to understand the impact of collected data across geographic locations, network carriers, and devices, MobiPerf allows a comparative study of past network measurements made by different users, but prevents users from running similar measurements to limit contention for testbed resources. New measurements are executed only if a query for previously collected data comes back empty. The latest version of MobiPerf, released in August 2014, is based on Mobilizer – an open-source Android library for network measurement announced at AIMS 2014 [41].

Functionality: MobiPerf supports several types of network performance measurement, which can execute serially or in parallel [17]. Mobilyzer provides measurement isolation (only one experiment is active at a time), which avoids bandwidth contention and radio power state transitions across experiments. To measure throughput, Mobiperf transmits random data to and from a nearby M-Lab server for 16 seconds and computes uplink and downlink throughput from packet traces.

MobiPerf supports latency measurements on both IPv4 and IPv6 network paths, using ICMP ping when available, with fallback to a Java ping implementation and latency estimates from three-way TCP handshakes in HTTP transfers. Mobiperf measures the delay of DNS lookups using the default DNS server configured for the device, which limits the ability to measure performance of third-party open DNS infrastructure.

MobiPerf also supports measurement of uplink and downlink UDP packet loss, out-of-order delivery, and variation of one-way latency. To obtain these metrics on the uplink, a client device sends a group of UDP packets to a nearby M-Lab server, where the server calculates network metrics from packet arrival time and order. The same transmission repeats from server to client to calculate downlink metrics.

MobiPerf performs more complex measurements to discover fine-grained network policies and their effect on data plane performance. For example, MobiPerf measures radio resource control (RRC) state information of cellular networks to estimate the impact on packet latency [113]. Finally, MobiPerf measurements can execute in the background to support long-term monitoring of network performance.

Data Collection: Similar to other measurement tools, the MobiPerf application collects performance data such as TCP uplink and download throughput, HTTP download latency and throughput, traceroutes, path latency, and DNS lookup delay. Researchers and vendors may want to know how variation in mobile hardware affects application performance, so MobiPerf collects device-related data such as manufacturer, model, operating system version, Android API level, carrier, salted hash of device IMEI, coarse-grained cell ID location information, cell tower ID and signal strength, Location Area Code (LAC), local IP address, IP address seen by the remote server, GPS coordinates, ports blocked by cellular provider and network connection type (HSPA/LTE) [114], [115].

Resource Incentives and Protection: MobiPerf relies on user curiosity to support measurement, and users can limit the resources they contribute. Specifically, measurements do not execute when the device battery consumption, or MobiPerf application monthly data usage, exceed user-set thresholds.

Privacy Protection: MobiPerf currently records the users' e-mail address, if they choose to provide one, to access their historical measurement results. This information is secured by Google's account authentication mechanisms and is not made publicly available. To minimize any

risk of exposing this potentially personally identifiable information, future versions of MobiPerf will store a salted hash of users' e-mail addresses instead.

Limitations: MobiPerf allows users to choose from only predefined measurements, which limits the tool flexibility. For example, MobiPerf does not support transfers of custom content on arbitrary ports to detect network traffic shaping.

4.2.2 ALICE

A Lightweight Interface for Controlled Experiments (ALICE) is a programmable network measurement library for Android devices developed by John Rula *et al.* at Northwestern University [50]. ALICE extends Dasu, a rule-based network testbed built as an add-on to the Vuze BitTorrent client [46], by enabling experiment definition in Javascript [116].

Functionality: The ALICE measurement library supports active and passive experiments on mobile devices. ALICE provides a programmable interface for the configuration of active network measurements, such as DNS resolution, ping, and iPerf. Tests can execute sequentially or in parallel. Although the sequence of tests and value passing between them is organized through a Javascript experiment definition, ALICE does not support custom traffic generation, and so is primarily a network measurement library. For serially scheduled experiments, ALICE allows one experiment on a device at a time; for parallel execution, ALICE allows a limited number of experiments to run at the same time – new experiments scheduled for a given device enter a queue until the device becomes available. ALICE chooses its test devices based on user-specified time of day, network provider, and network type (Wi-Fi/Cellular).

Data Collection: ALICE collects device location, radio signal strength (WiFi and cellular), WiFi access point name, device hardware address, IP address on each network interface, and number of bytes sent and received by other applications on the device. ALICE also collects performance metrics, including HTTP GET request time, DNS lookup time, ping times, available bandwidth. ALICE records network diagnostic information provided by traceroute and NDT (Section 3.3.1).

Resource Incentives and Protection: As of September 2014, ALICE has been included in three different applications developed at Northwestern University and available through the Google Play store: Namehelp Mobile[1], Application Time (AppT)[2], and NU Signals v2[3]. The Northwestern team's deployment model of growing the tool through application deployments allows ALICE to benefit from popularity spikes of new applications. To protect device resources, developers can set quotas for bandwidth usage of individual measurements.

Privacy Protection: ALICE records hardware addresses of available network interfaces, which are unique to each device. In combination with the ability to record sent and received traffic payload of other applications, for example location reporting, ALICE creates a potential for privacy exposure, if user location, or other private data, is correlated to unique device ID.

Remaining Challenges: Currently ALICE does not support repeatable experiments on the same device, or set of devices, through device selection criteria. ALICE also does not support peer-to-peer experiments, or custom traffic transmissions, which limits the tool's support for application prototyping.

5 Measurement Services

In addition to testbeds and tools there are many closed-source, proprietary measurement services for mobile networks. We divide these network monitoring and network discovery and diagnosis. The main goal of these is to collect data and provide insight to users based on their own device, but not necessarily make the data broadly available. Still, these services offer valuable insight to developers, researchers, regulators, and network operators able to access the data. Because the details of how these services are implemented and how they perform measurements is not widely available, we restrict our discussion, with few exceptions, to the commonalities and differences of what data these services collect.

5.1 Network Monitoring

Google Play Store and Apple App Store offer tens of applications for monitoring of network performance. Because of their relative similarity, we restrict our discussion to several popular and representative services.

5.1.1 Ookla SpeedTest Mobile

Ookla's SpeedTest application for mobile devices, released in January 2009, measures the device's network performance over Wi-Fi and cellular links [55]. As of March 2015, the application support measurements against 3479 geographically distributed Ookla servers in about 80% of world's ISP networks, has over 10 million installations, and has successfully completed over 7 billion user initiated measurements on the Ookla infrastructure [120], [121]. Ookla also allows users to host an Ookla server To expand the capacity of their measurement infrastructure, Ookla also allows users to host an Ookla server [122]. The application is available for Android, iOS, Windows phone, and Amazon FireOS based smartphones [55].

1. Namehelp Mobile measures the DNS performance of Cellular ISPs and public DNS resolvers, including of CDN replicas [117]
2. Application Time allows users to track their application usage on their mobile device [118].
3. NU Signlas allows users to diagnose Wi-Fi problems [119].

Functionality: The application captures the device geographic location and uses it to identify a set of five nearby servers. If the device location is not available from the GPS, the application uses device's IP address and estimates the device's location using MaxMind's (approximate) IP-to-location database [123]–[125]. After identifying a pool of five nearby servers, the application sends a `hello` message to all five servers and selects the measurement server from the first received reply [126]. Users may also select a specific server based on criteria such as hosting ISP, distance from user, and city name.

This SpeedTest uses HTTP fetches of small files to measure round-trip time and compute uplink and downlink throughput [127]. To measure the ping latency, the application sends several HTTP requests and records the time when app receives responses from the server [127]. Ookla SpeedTest uses the computed connection throughput to estimate how much data it can download from the server within 10 seconds, and then uses up to four HTTP threads on a single persistent connection to download the estimated amount of data. To eliminate any influence on the throughput results from protocol overhead, buffering time on the device, CPU usage, the application first discards the fastest and slowest 10% of throughput values as outliers before computing the average throughput. The application then discards the slowest 20% of throughput values to prevent results from being influenced by TCP slow-start. Finally, the application calculates the downlink throughput for the experiment based on the average of the remaining throughput values. The uplink throughput test is similar to downlink throughput test.

Data Collection: Ookla's SpeedTest mobile application collects device location (GPS and network-based), radio signal strength, device ID, device phone number, call status and remote phone number of an active call, names of devices on connected Wi-Fi network, local and public IP addresses, time at which the experiment was conducted, round-trip time, upload and download throughput, and connected network type (Wi-Fi or cellular). Ookla supports another application, *PingTest*, that collects network jitter and packet loss, to understand the suitability of the user's network for services such as VoIP audio, video streaming, and online gaming [128].

Resource Incentives and Protection: The application limits the number of HTTP threads to two when the observed throughput is less than 4 Mbps, otherwise the it uses four threads for throughput experiments.

Privacy Protection: The SpeedTest mobile application collects personally identifiable information (phone number, device ID, and device location), which may expose private information, such as a user's location when a measurement occurred. Users may delete previously collected data, or leave it on Ookla servers to compare with new data collected at a later time to discover changes in network performance over time.

Limitations: As of March 2015, the SpeedTest mobile application lacks a programming interface to allow users to automate and schedule experiments. Although, Ookla allows users to host SpeedTest experiments on their Web servers for in-house testing, via SpeedTest Mini, however, as of March 2015, the ability to run measurement against such servers is not supported on Speedtest's mobile application and is only supported with the Web version of Ookla SpeedTest [129]. The algorithm used by the application to measure the round-trip time relies on the time it takes to receive an HTTP response, which may include the time request spent in transport queue and application processing at the server. Finally, the application does not support detection of traffic shaping.

5.1.2 RadioOpt Traffic Monitor

The RadioOpt Traffic Monitor mobile application, released in April 2012 by RadioOpt GmbH, allows users to understand the performance, reliability, and utilization of their wireless and cellular networks [53]. Based on the information collected about the network, the application allows users to compare the performance of their wireless networks with other users in the same geographic region. The application is available for Android, iOS (iOS 7.0 or later), Blackberry, and Windows-based smartphones [130], [131]. As of March 2015, the application was installed over a million times.

Functionality: The RadioOpt Traffic Monitor mobile application uses CacheFly's CDN infrastructure. To identify a nearby server, the application sends a DNS query to the device's default DNS server for a CacheFly CDN domain name (`cdn2.speedtestsdk.com`). CacheFly uses TCP-anycast to direct users to the nearest CDN replicas [132]. Next, the application sequentially initiates downlink and uplink throughput tests to the selected server. To measure throughput, the application estimates the appropriate size of the data to exchange between the device and the server, similar to Ookla SpeedTest.

To measure latency, the application sends 15 ICMP ping requests to the server and records the time of each request/response pair. To measure the time to load a webpage on user's network, the application sends three HTTP GET requests and records the time to download the complete webpage, and other web objects such as CSS, image, JavaScript files embedded into the page.

Data Collection: The application computes parameters from measurement data such as the minimum, average, maximum ping latency to a nearby CacheFly server along with the standard deviation in latency and throughput, the amount of data uploaded and downloaded for the throughput tests, download time

of a hosted web page and the web page size. The application also collects device-related information such as its location (including accuracy and device travel speed), web bookmarks and browsing history, names of devices connected to the same Wi-Fi network, signal strengths at different locations, number of SMSes sent and received, and incoming and outgoing voice minutes, device model and manufacturer, OS or firmware version, current time on the device, the time when the device was last rebooted, cellular access technology (2G/3G/4G), and network country code.

The application also collects information specific to applications on the device such as their names, duration of usage, cellular and Wi-Fi data consumption (only on Android based smartphones), memory consumption, traffic (per application) sent and received on the device over cellular and Wi-Fi networks, application type (OS service or background), and software packages used by the application.

The application also collects device battery-specific information such as the battery state and charge remaining, voltage, temperature, technology, and charging state. Finally, the application collects Wi-Fi network related information such as the signal strength (latest, minimum, and maximum), network SSID and BSSIDs, the MAC address of the client, IP address of the client, and client-to-router link bandwidth.

Resource Incentives and Protection: RadioOpt relies on user curiosity to understand the performance of their own wireless and cellular networks. The app allows users to configure a monthly/weekly/daily cellular data cap, monitor their monthly data traffic, SMSes received and sent, and voice minutes, and configure alerts when data, SMS, or voice minutes reach a threshold.

Privacy Protection: The application may discover user behavior since it collects information such as the user's Web browsing history, bookmarks, applications installed and their duration of usage, among others. However, any personally identifiable data collected by RadioOpt mobile application is not shared with RadioOpt servers without the user's consent.

Limitations: RadioOpt does not allow its users to understand whether their cellular ISPs are discriminating one traffic over the other. Further, the application does not support measurement experiments to be run against an arbitrary server.

5.1.3 OpenSignal

The OpenSignal mobile application, released in March 2013 by OpenSignal, Inc., allows users to compare the quality and coverage of their cellular networks (on a Google Map's developer widget [133]) in different geographic areas and with other cellular networks available in the area [51]. The application rates for how well Web, Video, and VoIP based applications are likely to perform on the current cellular network. The application assist users to also find publicly available free and paid Wi-Fi hot-spots, and the walking directions for higher signal strength. The application has over 10 million installations and is available for Android and iOS based smartphones [134], [135]. As of March 2015, the application has garnered over 900,000 users and has performed several network measurements to collect information for over 800,000 cellular towers, 825 cellular networks, over 5B cellular signal readings, and over 1B Wi-Fi access points available in different countries [51].

Functionality: The OpenSignal mobile application supports several active and passive measurements to measure ping latency, download and upload throughput. The application performs periodic passive measurements, and publishes them to an OpenSignal server [136]. Before starting any measurement test, the application sends the device ID, OS, Android API version, and BSSIDs of nearby wireless networks to an OpenSignal server. Next, to measure latency, the application sends 3 HTTP HEAD requests to `www.google.com` [137], [138]. The application then records the time to receive the time to get the response for each request, followed by calculating the average of the three latency values.

To measure the download throughput, the application sends eight concurrent HTTP GET requests to download files of size 108 Mb each, from a CloudFront's CDN replica [136]. The download throughput test is performed for a fixed amount of time after which the application computes the average throughput. To measure the upload throughput, the application sends several concurrent HTTPS POST requests to upload several small image files of size 15 Mb in total, to an Amazon AWS server.

As a part of making the collected data available publicly and to encourage developers, researchers, regulators, and network operators to investigate and address network problems, OpenSignal provides two APIs [139]. The first API, known as NetworkStatus, allows developers to get signal strength, upload and download throughput, round trip latency, and network name, network ID, network type (2G/3G/4G), and network reliability for every measurement within certain distance of a specified geographic coordinate [140]. The second API, known as Tower Info, allows developers to get the cell ID, location area code, phone type (GSM/CDMA), and estimated latitude and longitude of a cellular tower [141]. To prevent misuse of their publicly available API, OpenSignal allows a maximum of five API calls every minute and 2000 API calls every month.

Data Collection: The OpenSignal mobile application collects device-related information such as SMS transmission and receipt timestamps, device location, ID, model name, OS, Android API level, IP address, behavior at different battery temperatures (hot, crashed, slow, fast),

duration of OpenSignal sessions on the device, and whether the phone is engaged in a phone call during the measurement.

The application collects network-related information such as the active Wi-Fi SSID, names of devices connected to the Wi-Fi, SSIDs of other avaiable Wi-Fi, connection type (collected every 15 minutes), signal strength, upload and download throughput, and round trip latency to a Google server. For devices connected to GSM networks, the application associates cell towers by their cell id and location area code; for CDMA networks, by their Network ID, Base sub-station ID and system ID [134]. To understand the relationship between signal quality and battery consumption, the OpenSignal application collects the battery level, voltage and temperature [142].

Resource Incentives and Protection: OpenSignal does not provide any incentives for user participation to run measurement experiments on mobile devices.

Privacy Protection: Although the application collects information about phone calls and SMS messages, the application never reads them [143]. This is because the application only counts the total number of text messages received and sent from the device. Further, any personally identifiable information collected by the OpenSignal mobile application is never shared by any third party services [143]. However, OpenSignal does not take any responisbility of any data shared by the user on online social networking websites through the OpenSignal application. Finally, the application does not put any obligation on the user to share the data collected with OpenSignal.

Limitations: The application does not detect the presence of traffic shaping in ISP networks. Further, to perform throughput measurement tests, the application requires an exchange of several hundred of megabytes between the mobile device and the server, which may not be suitable for users with low data plans [136].

5.1.4 Vodafone NetPerform

The Vodafone NetPerform mobile application, released in June 2014 by Vodafone Sales and Services Limited, allows users to understand the performance of their cellular network in their region and compare with it with the performance that other users in the same region are experiencing [144]. The application also allows Vodafone to understand the amount of data that their customers use and as well as the trend in data usage by tracking the data usage from different applications installed on their customers' smartphones. Such knowledge of data usage allows Vodafone to resolve connectivity issues in their network, as well as, install higher capacity links to accommodate any customer demands to support interactive applications that require higher bandwidth. The data used by the Vodafone NetPerform mobile application is free for only Vodafone customers in Ghana, Ireland, and United Kingdom. However, users in other countries or non Vodafone customers may be charged for any data used by the Vodafone NetPerform application. The application is available for Android and iOS based smartphones [145], [146].

Functionality: Every hour, the application establishes a TCP connection with a Vodafone server to verify whether the device has Internet connectivity. Conducting such a test every hour allows Vodafone to understand the network stability and any variation in end-to-end latency on their network over time. The application performs another hourly network measurement test to determine the uplink and downlink throughput against a nearby Vodafone server. The throughput tests execute for only 10 seconds, within which the application exchanges data with a Vodafone server [145]. The throughput is then calculated as the average of different throughput values sampled in 10 seconds.

Data Collection: The data collected by the Vodafone NetPerform mobile application is stored on Vodafone servers for only 14 months, which allows Vodafone to understand the changes in the seasonal use of the network usage by their customers. To understand and diagnose the network problems related to phone call connectivity, the application collects cellular tower ID to which the device is connected, signal strength, device location when the network is either limited or not available, the quality of 2G/3G coverage, device speed (if available through GPS), and time duration when the device uses cellular network, how the phone call ends (dropped or disconnected by the user) [147].

To understand and diagnose issues related to data services the application additionally captures whether the device can establish a connection with a Vodafone server, time taken to establish a connection with a Vodafone server, the MAC addresses of all available Wi-Fi access points along with their link bandwidth, hourly data usage of the device, data usage when the device is in standby mode, and the upload and download throughput [147].

To understand the types of Internet services that users are interested in and to allocate high capacity bandwidth for services that require high bandwidth, the application captures the names of all applications installed on the device, the names of applications that the user uses everyday, the duration of application use, the amount of data is received and sent from each installed application.

Finally, to diagnose and resolve device related network issues, the application collects the device model and company, device IMEI (encrypted to maintain anonymity), the OS running on the device, firmware version, the OS language, battery status, memory in use, the time when the phone last rebooted [147].

Resource Incentives and Protection: Users do not get any incentives for running measurement tests on their devices. Instead, Vodafone relies on users' curiosity to understand the network performance and gathers data collected on users' devices to improve the quality of their voice and data services. With respect to protecting device resources, the application does not allow users to configure a monthly cap on the amount of cellular and Wi-Fi data that the application can use to run measurement tests. Further, since the application run throughput and latency tests every hour, the application prevents the device to turn off its radio, which drains the device's battery quickly [145].

Privacy Protection: The application does not collect any personally identifiable information such as the device phone number, the phone numbers of incoming and outgoing phone calls, incoming and outgoing SMS messages, and the names of available Wi-Fi hotspots. However, by collecting the names of application installed and when different applications are used, the Vodafone NetPerform application has a potential to discover user behavior, which might be unsuitable for some users.

Limitations: The Vodafone NetPerform mobile application does not allow users to discover whether their cellular ISPs are performaing traffic discrimination. Further, the availability of the application only for users in a few countries in Europe restricts the network operators in other countries to gain insight of their network issues and performance.

5.1.5 Emerging Applications

Many other network measurement services have been developed by independent developers to assist users to measure performance of wireless and cellular networks. Such applications include SpeedSpot [148], Sensorly [149], RootMetrics [150], NetworkCoverage [151], Internet Speed Test [152], Netradar [153], Cisco Data Meter [154], 4Gmark [155], and nPerf [156]. Because the details of how these services are implemented and how they perform measurements is not widely available, we restrict our discussion to the commonalities and differences of what data these tools collect and how. We also illustrate the similarities and differences between these emerging measurement services in Table 2.

Specifically, SpeedSpot, Sensorly, RootMetrics, and NetworkCoverage are similar to the OpenSignal mobile application in their capability for users to compare the performance of their wireless and cellular networks on a map and find nearby Wi-Fi networks. Internet Speed Test is similar to Ookla SpeedTest; it allows users to measure the latency and throughput to application's servers on user's network. Similar to RadioOpt, NetRadar and Cisco Data Meter applications run latency and throughput experiments against servers deployed on the cloud/CDN servers and allows users to monitor traffic sent and received by applications installed on their devices. 4Gmark and nPerf are similar to each other, in that, these applications not only allow users to measure the performance of network in terms of throughput and latency, but also measures the suitability and reliability of the network for streaming and Web applications.

5.2 Network Discovery and Diagnosis

While most of the previous projects focus on measuring end-to-end performance of mobile application communications, the following tools allow developers and researchers to learn more about the state of network infrastructure and configurations that affect transmission of application traffic. Pertinent features include the presence of proxy servers and other middleboxes, or complex multi-level DNS resolutions.

5.2.1 NDT (Mobile Client)

The Network Diagnostic Test (NDT) system, developed by Internet2, evaluates the performance of mobile connections to diagnose problems that limit network bandwidth [42], [157]. NDT also detects problems associated with device misconfiguration and network infrastructure. NDT (Mobile) is currently hosted on Google's M-Lab and allows access to its backend through an Android mobile application.

Functionality: NDT measurements are performed from a mobile Web browser that issues requests to NDT servers, hosted by M-Lab. The server-specific tests diagnose observed network problems. After the measurement experiment completes, the server analyzes the results and returns them to the client device.

Data Collection: The NDT mobile application collects traffic performance information such as upload and download speed, round trip network latency (minimum, average, and maximum), jitter, TCP receive window size (current and maximum), packet loss, TCP retransmission timer, and number of selective acknowledgements received. The application also detects router cable faults, incorrectly set TCP buffers in the device, duplex mismatch conditions on Ethernet links, presence of NAT, and capacity limits.

Resource Incentives and Protection: The incentive model for the NDT mobile client is based on providing network diagnostic information in exchange for users running tests on their mobile devices. One issue for users who volunteer their device resources is that NDT requires permission to prevent the phone from going into power save mode, which may drain the battery quickly.

Privacy Protection: By default NDT records experimental data separately for each user, which allows users to privately diagnose their network problems. Data isolation also prevents malicious users from learning of open ports and interfaces in others' networks.

	SpeedSpot	Sensorly	RootMetrics	NetworkCoverage	Internet SpeedTest	NetRadar	Cisco Data Meter	4GMark	nPerf
Support measurement tests									
Uplink throughput	✓	✓	✓		✓	✓	✓	✓	✓
Downlink throughput	✓	✓	✓	✓	✓	✓	✓	✓	✓
Latency	✓	✓		✓	✓	✓	✓	✓	✓
Signal coverage maps		✓	✓	✓		✓			
Uplink throughput tests									
Total uplink transfer	6 MB	10 MB	7 MB		8 MB	Random	900 KB	50 MB	20 MB
Probe method	P	P	P		P	TCP	P	P	P
No. of probes	1	2	1		12	Random	1	1	20
Duration	6 s	U	8 s		15 s	10 s	U	10 s	U
Downlink throughput tests									
Total downlink transfer	20 MB	400 MB	2400 MB	10 MB	400 MB	1 MB	2 MB	250 MB	10 GB
Probe method	G	G	G	G	G	TCP	G	G	G
No. of Probes	2	4	4	1	4	16	2	1	10
Duration	U	10 s	25 s	U	15 s	10 s	U	10 s	10 s
Latency tests									
Probe Method	G	G		T	P	G	T	G	G
No. of probes	10	6		Random	Random	2	30	3	10
Measure Application performance									
Webpage load time								✓	✓
Throughput of video streams								✓	✓
Application deployment									
Experiments run against servers hosted by	MaxCDN, EdgeCast CDNs	OVH, Digital Ocean CDNs	Amazon AWS	Think Broadband, Emanics Lab	v-speed	Amazon AWS, CacheFly CDN	Akamai CDN	4Gmark	nperf
Supported mobile OS platform	Android, iOS	Android, iOS	Android, iOS	Android	Android	Android, iOS, Windows, MeeGo, Symbian, NokiaX, Jolla, and Blackberry	Android, iOS	Android, iOS	Android, iOS
Misc.									
Developed by	Speed Spot	Sensorly	Root Metrics	Technische Universitt Darmstadt	V-Speed	Aalto University	Cisco Systems	Trois Petits Points	nPerf
Released in	May 2013	Aug. 2012	Aug. 2011	Mar. 2015	Oct. 2014	Feb. 2013	May 2013	May 2013	Oct. 2014
Number of app installs	> 100 K	> 50 K	> 100 K	> 100	> 1 M	> 10 K	> 50K	> 500 K	> 50 K

Legend:
U – Until transfer completes.
G – HTTP GET.
P – HTTP POST.
T – TCP Connection Setup.

TABLE 2
Experimentation flexibility matrix of emerging end-to-end measurement services.

Limitations: The NDT mobile client executes experiments that evaluate network traffic only between a mobile device and its closest M-Lab server and not any arbitrary server.

5.2.2 Netalyzr

The Netalyzr mobile application, developed as a collaboration of ICSI Berkeley, UC Berkeley, HIIT, and Aalto University, is a diagnostic tool that characterizes connectivity, performance anomalies, and network security issues [43], [158]. The tools measures network latency and bandwidth to reveal insight into not only performance to cloud servers, but also how middleboxes in the path affect the performance of traffic. As of March 2014, Netalyzr has run over 15000 times to diagnose 290 operators in 90 countries. Netalyzr is accessible via an Android mobile application available on the Google Play Store.

Functionality: Netalyzr identifies the presence of Network Address Translations (NATs), proxy servers along a route, IP fragmentation, size of bottleneck buffers, reachability of services, and presence of HTTP proxies. When the Netalyzr application starts, it contacts the Netalyzr's Web server, which issues a DNS lookup request to redirect the user's request randomly to one of the twenty Netalyzr's back-end servers hosted on the Amazon cloud. Each back-end server supports twelve concurrent measurement sessions.

Netalyzr detects the presence of a NAT based on a difference between a user's local and public IP addresses. For clients behind a NAT, Netalyzr identifies how the network renumbers addresses and ports, i.e., whether the NAT uses fixed associations of local IP addresses to different public IP addresses, or if the NAT uses load-balancing.

To detect support for IP fragmentation, Netalyzr sends a 2 KB UDP packet (larger than 1500 B Ethernet maximum transmission unit (MTU)) to the server – a response from the server indicates the network supports fragmentation. If there is no response Netalyzr uses binary search to find the maximum packet size it can deliver without the packet being fragmented at the IP layer. The same test repeats from server to client to detect network support for fragmentation on the reverse path.

The sizing of bottleneck buffers affects user-perceived latency, and is measured based on the difference in latency during inactivity and during path throughput tests. Finally, queue drain time indicates the size of the buffer. To perform service reachability related experiments, the application attempts to connect to 25 different well known ports on a back-end server.

Netalyzr infers the presence of HTTP proxies if the public IP address in the request received by the back-end server is not the same as the client's public IP address. To detect the presence of in-path HTTP proxy, the client first sends an HTTP request to the server, the server then returns the request headers it received in the request back to the client. The client then compares the headers it sent and the headers the server sent to the client for any added, deleted or modified fields. To detect the presence of caching policies, the application relies on the HTTP 304 Not Modified response from the server.

To detect the presence of a DNS-proxy server or firewall, the application sends a DNS request to Netalyzr's back-end server. If the client detects any change in the response (different transaction ID, or public IP address), then Netalyzr assumes an in-path DNS proxy exists. Netalyzr then makes invalid DNS requests to the back-end server. If the client receives an invalid response from the server, nothing is detected, but if the request is blocked, Netalyzer assumes a DNS-aware middlebox is blocking invalid DNS requests from leaving the network.

Data Collection: The Netalyzr mobile app records the presence of network interfaces, gateways, NAT detection, port renumbering, path MTU, packet fragmentation, DNS resolver, extension mechanisms for DNS (EDNS) support, port randomization, IPv6 support, hidden proxies, in-path caches, header manipulation, image transcoding, compression, HTTP type filtering, port filtering, traffic differentiation, IP fragmentation, signal-to-noise ratio, Wi-Fi/cellular configuration, network topology through traceroute, TLS handshake, UPnP vulnerabilities on Wi-Fi APs, clock drift, and TLS default certificates [158].

Resource Incentives and Protection: Netalyzr provides network diagnostic and troubleshooting information to users. Netalyzr requests user permission to modify system settings and to terminate other running applications in order to increase measurement accuracy. The Netalyzr mobile application asks users for permission to execute IP traceroutes, since ICMP packet transmission on a mobile device requires access to raw sockets.

Privacy Protection: Netalyzr asks users to opt in to the data collection process before installing the application. Therefore, if users are uncomfortable with sharing the measurement results with Netalyzr, they may not install the application. However, when the user grants permissions to the application, Netalyzr could use GPS to get device location, read phone status and identity, and modify or delete the contents of USB storage to store or delete measurement related data on the device.

Limitations: Although Netalyzr provides a robust diagnostic set of end-to-end network measurements and helps users troubleshoot networks, unlike MITATE, or WindRider, Netalyzr does not detect traffic shaping in mobile ISPs.

5.2.3 PortoLan

PortoLan is a network experiment testbed based on volunteered mobile devices that executes experiments submitted to back-end servers [159]. PortoLan is designed by Enrico Gregori *et al.* at Istituto di Informatica e Telematica, to discover Internet topology and build wide scale mobile network signal quality maps. The Android application for PortoLan is available on Google Play and allows users to run measurement tests like ping, traceroutes, maximum throughput, and detection of traffic shaping of BitTorrent traffic [160]. The PortoLan team intends to add capability to support active network experiments and access to mobile sensor data such as network signal strength, device location, network name, cell type, and roaming status. PortoLan relies on user altruism to build testbed capacity and support measurement. The PortoLan mobile application limits the device cellular bandwidth usage to 2 MB/day and postpones experimentation when battery drops below 40%. Finally, the application does not collect personally identifiable information from the device and anonymously stores measurement data on backend servers.

6 Conclusions

This survey provides a comprehensive overview of the existing and emerging end-to-end mobile network measurement testbeds, tools, and services. In spite of the relative maturity of existing platforms, several functionality gaps remain with respect to the needs of developers, researchers, network operators, and regulators in assessing mobile network performance. First, existing tools do not adequately support detection of traffic shaping. As depicted in Table 1, testbeds such as MITATE, Seattle, PhoneLab, PortoLan, and WindRider can detect the presence of traffic shaping mechanisms in mobile ISPs, whereas, other testbeds do not. Second, device churn inherent in platforms based on ad-hoc user participation means that existing tools are not well-suited for long-term network performance monitoring. In fact, the popularity of tools such as PhoneLab and FCC has declined over time. Third, several testbeds enable developers to prototype the performance of their applications ahead of deployment. However there is significant disparity in how testbeds provide that functionality in terms of execution models and APIs. Finally, exchange of P2P traffic, network diagnostics, ICMP traceroutes, device selection criteria, and NAT traversal are not only selectively supported by different platforms. One significant axis of comparison between network measurement platforms not discussed in this survey is their accuracy. The variety of measurement methods used to obtain even the relatively standard network metrics, such as throughput, makes it difficult to compare the relative accuracy of the different platforms.

Based on the surveyed work, we believe the mobile network measurement community needs a more concerted effort among developers, researchers, network operators, and regulators to produce network measurement tools that meet the needs of all four communities. A more concerted effort would lead to greater adoption of (perhaps fewer) tools, as well as large-scale and long-term network monitoring. At the same time, funding agencies should support development of new measurement approaches and capabilities, especially when such improvements are aimed at enhancement of existing testbeds.

Acknowledgments

The authors would like to thank Justin Cappos, Geoffrey Challen, David Choffness, Nick Feamster, Walter Johnston, Valerio Luconi, Jim Martin, Konstantina Papagiannaki, John Rula, Mario Sanchez, Qing Yang, and Hongyi Yao for suggested improvements to an early version of this manuscript.

References

[1] "USTREAM," http://www.ustream.tv/, Jul. 2014.
[2] J. Codorniou, "Whats next for social mobile games?" http://techcrunch.com/2012/12/22/whats-next-for-social-mobile-games/, Dec. 2012.
[3] K. Rogers, "What's next after WhatsApp: a guide to the future of messaging apps," http://www.theguardian.com/technology/2014/feb/21/whatsapp-facebook-messaging-apps-viber-kik, Feb. 2014.
[4] P. Marupaka, "The future looks bright for augmented reality," http://www.siggraph.org/discover/news/future-looks-bright-augmented-reality, May 2014.
[5] K. Shubber, "Microsoft kinect used to live-translate sign language into text," http://www.wired.co.uk/news/archive/2013-07/18/sign-language-translation-kinect, Jul. 2013.
[6] I. Rimington, "Leave your wallet at home and pay with your profile picture," https://www.paypal.co.uk/Blog/Leave-your-wallet-at-home-and-pay-with-your-profile-picture/, Aug. 2013.
[7] I. Grigorik, *High Performance Browser Networking*. O'Reilly, 2013.
[8] C. Zhang, C. Huang, P. A. Chou, J. Li, S. Mehrotra, K. W. Ross, H. Chen, F. Livni, and J. Thaler, "Pangolin: speeding up concurrent messaging for cloud-based social gaming," in *ACM CoNEXT*, December 2011.
[9] S. Agarwal and J. R. Lorch, "Matchmaking for online games and other latency-sensitive P2P systems," in *ACM SIGCOMM*, August 2009.
[10] J. Erman, V. Gopalakrishnan, R. Jana, and K. K. Ramakrishnan, "Towards a SPDY'Ier Mobile Web?" in *ACM CoNEXT*, Dec. 2013.
[11] J. Butler, W. Lee, B. McQuade, and K. Mixter, "A Proposal for Shared Dictionary Compression over HTTP," http://lists.w3.org/Archives/Public/ietf-http-wg/2008JulSep/att-0441/Shared_Dictionary_Compression_over_HTTP.pdf, Sep. 2008.
[12] Akamai, "Free Mobile Web Performance Measurement Tool," http://mobitest.akamai.com/m/index.cgi, 2012.
[13] "Perfecto Mobile," http://www.perfectomobile.com/, 2014.
[14] W. Johnston, "Measuring Broadband America," http://www.caida.org/workshops/aims/1403/slides/aims1403_wjohnston.pdf, Mar. 2013.
[15] Readwrite, "Net Neutrality: What Happens Now That Verizon Has Vanquished The FCC," http://readwrite.com/2014/01/15/net-neutrality-fcc-verizon-open-internet-order, Jan 2014.
[16] B. Zevenbergen, I. Brown, J. Wright, and D. Erdos, "Ethical Privacy Guidelines for Mobile Connectivity Measurements," http://www.oii.ox.ac.uk/research/projects/?id=107, 2013.
[17] J. Huang, C. Chen, Y. Pei, Z. Wang, Z. Qian, F. Qian, B. Tiwana, Q. Xu, Z. M. Mao, M. Zhang, and P. Bahlc, "MobiPerf: Mobile Network Measurement System (Technical report)," University of Michigan and Microsoft Research, Tech. Rep., 2011.
[18] k. claffy, D. D. Clark, and M. P. Wittie, "The 6th Workshop on Active Internet Measurements (AIMS-6) Report," *Sigcomm CCR*, October 2014.
[19] "What is Measurement Lab?" http://www.measurementlab.net/about, 2014.
[20] U. Goel, A. Miyyapuram, M. P. Wittie, and Q. Yang, "MITATE: Mobile Internet Testbed for Application Traffic Experimentation," in *Mobile and Ubiquitous Systems: Computing, Networking and Services (Mobiquitous)*, Dec. 2013.
[21] K. Winstein, A. Sivaraman, and H. Balakrishnan, "Stochastic forecasts achieve high throughput and low delay over cellular networks," in *USENIX NSDI*, Apr. 2013.
[22] S. Higginbotham, "Traffic Shaping Coming to a Mobile Network Near You," https://gigaom.com/2011/04/05/traffic-shaping-coming-to-a-mobile-network-near-you/, Apr 2011.
[23] M. Marcon, M. Dischinger, K. P. Gummadi, and A. Vahdat, "The Local and Global Effects of Traffic Shaping in the Internet," in *Communication Systems and Networks (COMSNETS)*, Jan 2011.
[24] E. Howard, C. Cooper, M. P. Wittie, S. Swinford, and Q. Yang, "Cascading impact of lag on user experience in multiplayer games," in *ACM NetGames*, December 2014.
[25] YUI Team, "Combo Handler Service Available for Yahoo-hosted JS," http://yuiblog.com/blog/2008/07/16/combohandler/, Jul. 2008.
[26] Jeff Barr, "Multi-Region Latency Based Routing now Available for AWS," https://aws.amazon.com/blogs/aws/latency-based-multi-region-routing-now-available-for-aws/, Mar. 2012.
[27] Matthew Prince, "Mirage 2.0: Solving the Mobile Browsing Speed Challenge," https://blog.cloudflare.com/mirage2-solving-mobile-speed/, Jun. 2013.
[28] Opera Software ASA, "Opera Turbo," http://www.opera.com/turbo, Mar. 2015.
[29] "Apptimize," http://apptimize.com/, May 2015.

[30] "Splitforce," http://splitforce.com/, May 2015.
[31] "Optimizely," http://www.optimizely.com/, May 2015.
[32] "Google Analytics," http://www.google.com/analytics/, Sep. 2014.
[33] Y. Zhuang, A. Rafetseder, and J. Cappos, "Experience with Seattle: A Community Platform for Research and Education," in *GENI Research and Educational Workshop (GREE)*, Mar. 2013.
[34] A. Nandugudi, A. Maiti, T. Ki, F. Bulut, M. Demirbas, T. Kosar, C. Qiao, S. Y. Ko, and G. Challen, "PhoneLab: A Large Programmable Smartphone Testbed," in *Workshop on Sensing and Big Data Mining*, Nov. 2013.
[35] A. Faggiani, E. Gregori, L. Lenzini, V. Luconi, and A. Vecchio, "Network sensing through smartphone-based crowdsourcing," in *Embedded Networked Sensor Systems (SenSys)*, Nov. 2013.
[36] A. Striegel, S. Liu, L. Meng, C. Poellabauer, D. Hachen, and O. Lizardo, "Lessons learned from the netsense smartphone study," in *ACM Workshop on HotPlanet*, ser. HotPlanet '13, Aug. 2013.
[37] R. K. Balan, A. Misra, and Y. Lee, "Livelabs: Building an in-situ real-time mobile experimentation testbed," in *Workshop on Mobile Computing Systems and Applications (HotMobile)*, Feb. 2014.
[38] MobiPerf, "Welcome to MobiPerf," http://www.mobiperf.com/, 2014.
[39] I. Trestian, R. Potharaju, and A. Kuzmanovic, "Closing the Loop: Feedback at Your Fingertips," http://www.cs.northwestern.edu/~ict992/docs/draft.pdf, 2009.
[40] K. Claffy, "The 5th Workshop on Active Internet Measurements (AIMS-5) Report," in *ACM SIGCOMM Computer Communication Review, Volume 43, Number 3*, July 2013.
[41] H. Yao, A. Nikravesh, Y. Jia, D. R. Choffnes, and Z. M. Mao, "Mobilyzer: A Network Measurement Library for Android Platform," in *Workshop on Active Internet Measurements (AIMS)*, Mar 2014.
[42] MLAB, "NDT (Mobile Client)," http://www.measurementlab.net/tools/ndt-mobile, 2013.
[43] C. Kreibich, N. Weaver, B. Nechaev, and V. Paxson, "Netalyzr: Illuminating the edge network," in *ACM SIGCOMM Conference on Internet Measurement*, Nov. 2010.
[44] "SciWiNet," http://sciwinet.org/, 2014.
[45] K. V. d. Merwe, "PhantomNet: An end-to-end mobile network testbed," http://www.caida.org/workshops/aims/1403/slides/aims1403_jvandermerwe.pdf, Mar. 2014.
[46] M. A. Sánchez, J. S. Otto, Z. S. Bischof, D. R. Choffnes, F. E. Bustamante, B. Krishnamurthy, and W. Willinger, "Dasu: Pushing experiments to the Internet's edge," in *USENIX NSDI*, Apr. 2013.
[47] MLAB, "WindRider," http://www.measurementlab.net/tools/windrider, 2009.
[48] K. P. Gummadi, S. Saroiu, and S. D. Gribble, "King: Estimating latency between arbitrary Internet end hosts," in *SIGCOMM Workshop on Internet Measurement*, Nov. 2002.
[49] "M-Lab Tests," http://www.measurementlab.net/tests, 2014.
[50] J. Rula, "ALICE - Mobile Experiment Engine," http://aqualab.cs.northwestern.edu/projects/alice, Aug. 2014.
[51] OpenSignal, Inc., "OpenSignal," http://opensignal.com/, 2014.
[52] FCC, "Measuring Broadband America," http://www.fcc.gov/measuring-broadband-america/mobile, Nov 2013.
[53] RadioOpt GmbH, "RadioOpt," https://www.radioopt.com/, Mar. 2015.
[54] WindRider, "WindRider A Mobile Network Neutrality Monitoring System," http://www.cs.northwestern.edu/~ict992/mobile.htm, Oct 2009.
[55] Ookla, "Ookla SpeedTest Mobile Apps," http://www.speedtest.net/mobile/, 2014.
[56] "CRAWDAD: A Community Resource for Archiving Wireless Data At Dartmouth," http://crawdad.org/, Jan. 2015.
[57] "M-Lab," http://www.measurementlab.net/, Mar. 2015.
[58] "UMass Trace Repository," http://traces.cs.umass.edu/, Dec. 2009.
[59] J. Brodkin, "FCC votes for net neutrality, a ban on paid fast lanes, and Title II," http://arstechnica.com/business/2015/02/fcc-votes-for-net-neutrality-a-ban-on-paid-fast-lanes-and-title-ii/, Feb. 2015.
[60] J. Vijayan, "ATT, Sprint confirm use of Carrier IQ software on handsets," http://www.computerworld.com/article/2499667/application-security/at-t--sprint-confirm-use-of-carrier-iq-software-on-handsets.html, Dec. 2011.
[61] Carrier IQ, "Vodafone Portugal Pioneers Innovative Mobile Broadband Experience Management Architecture Using Carrier IQ Technology ," http://carrieriq.com/wp-content/uploads/2014/08/PR.CarrierIQandVodafonePortugal.20090730.pdf, 2009 Jul.
[62] M. Peckham, "Carrier IQ Wiretap Debacle: Much Ado About Something?" http://techland.time.com/2011/12/01/carrieriq-wiretap-debacle-much-ado-about-something/, Dec. 2011.
[63] "The 'secret' app installed on millions of mobile phones that records your keystrokes, your browsing and reads your messages," http://www.dailymail.co.uk/sciencetech/article-2068225/Secret-app-installed-millions-Android-phones-reads-messages.html, Dec. 2011.
[64] Lookout Labs, "CarrierIQ Scanner & Protection," https://play.google.com/store/apps/details?id=com.lookout.carrieriqdetector&hl=en, May 2013.
[65] sn707, "ATT LG G3 Carrier IQ Removal Guide," http://forum.xda-developers.com/att-lg-g3/general/att-lg-g3-carrier-iq-removal-guide-t2819295, Jul. 2014.
[66] V. Gopalakrishnan, L. E. Li, G. Ricart, J. Breen, J. Martin, Y. Xin, C. Elliott, A. Banerjee, J. Cho, M. Munakami, A. Chowdhary, N. Alsrehin, I. Alsmadi, D. Grunwald, E. Eide, R. Ricci, and M. Wittie, "SDN and NFV Report-Out," https://phantomnet.org/workshop/sdnnfv.pdf, Feb. 2015.
[67] C. Osborne, "The state of LTE 4G networks worldwide in 2014 and the poor performance of the US," http://www.zdnet.com/the-state-of-lte-4g-networks-worldwide-in-2014-and-the-poor-performance-of-the-us-7000026594/, Feb. 2014.
[68] J.D. Power, "Overall wireless network problem rates differ considerably based on type of service," http://www.jdpower.com/press-releases/2013-us-wireless-network-quality-performance-study-volume-2, Aug. 2013.
[69] T. Karr, "Verizon's Plan to Break the Internet," http://www.savetheinternet.com/blog/2013/09/18/verizons-plan-break-internet, Sept 2013.
[70] C. Aaron, "Net Neutrality Is Dead. Here's How to Get It Back," http://www.savetheinternet.com/blog/2014/01/14/net-neutrality-dead-heres-how-get-it-back, Jan 2014.
[71] M. Fahey, "Why Gamers Should Care About Net Neutrality," http://kotaku.com/5512448/why-gamers-should-care-about-net-neutrality, Apr 2010.
[72] S. Buckley, "Cogent and Orange France fight over interconnection issues," http://www.fiercetelecom.com/story/cogent-and-orange-france-fight-over-interconnection-issues/2011-08-31, Aug 2011.
[73] ——, "France Telecom and Google entangled in peering fight," http://www.fiercetelecom.com/story/france-telecom-and-google-entangled-peering-fight/2013-01-07, Jan 2013.
[74] A. Lynn, "Cable Companies' Big Internet Swindle," http://www.freepress.net/blog/2009/11/24, Nov 2009.
[75] N. Anderson, "Huge ISPs want per-GB payments from Netflix, YouTube," http://arstechnica.com/tech-policy/2011/01/huge-isps-want-per-gb-payments-from-netflix-youtube/, Jan 2011.
[76] R. Singel, "Mobile Carriers Dream of Charging per Page," http://www.wired.com/business/2010/12/carriers-net-neutrality-tiers/2/, Dec 2010.
[77] B. Cohen, "Incentives build robustness in BitTorrent," in *Workshop on Peer-to-Peer Systems (IPTPS)*, Feb. 2003.
[78] IBM, "CPLEX optimizer," www.ibm.com/software/commerce/optimization/cplex-optimizer/, 2013.
[79] MITATE, "MITATE : Mobile Internet Testbed for Application Traffic Experimentation (User Manual)," http://mitate.cs.montana.edu/sample/MITATE_Documentation_v1.0.pdf, Nov 2013.
[80] J. Cappos, I. Beschastnikh, A. Krishnamurthy, and T. Anderson, "Seattle: a platform for educational cloud computing," in *ACM SIGCSE Bulletin*, Mar 2009.
[81] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman, "PlanetLab: an overlay testbed for broad-coverage services," *SIGCOMM CCR*, vol. 33, no. 3, pp. 3–12, Jul. 2003.
[82] C. Siaterlis, A. Garcia, and B. Genge, "On the use of emulab testbeds for scientifically rigorous experiments," *IEEE Communications Surveys Tutorials*, vol. 15, no. 2, pp. 929–942, Feb. 2013.
[83] "Geni," http://www.geni.net/, Oct. 2012.

[84] yanyan, "Using Sensors In Seattle," https://seattle.poly.edu/wiki/UsingSensors, Apr 2012.
[85] "Sensibility testbed," http://sensibilitytestbed.com, Jul. 2014.
[86] PhoneLab, "PhoneLab Experimenter Agreement," http://experiment.phone-lab.org/terms/, 2013.
[87] PhoneLab, "PhoneLab A Programmable Smartphone Testbed," http://www.phone-lab.org/, 2013.
[88] ——, "Overview," http://participate.phone-lab.org/info/, 2013.
[89] "Devices Supported by SciWiNet," http://sciwinet.org/SciWiNet-Devices.html, 2014.
[90] LiveLabs, "Participation," http://livelabs.smu.edu.sg/participant/, Mar. 2013.
[91] "LiveLabs Registration," http://athena.smu.edu.sg/livelabs_register/, Oct 2014.
[92] FCCAPPs, "FCC Speed Test App," https://play.google.com/store/apps/details?id=com.samknows.fcc&hl=en, Dec 2013.
[93] J. Clover, "FCC Launches 'FCC Speed Test' iPhone App to Measure Mobile Broadband Performance," http://www.macrumors.com/2014/02/25/fcc-speed-test/, Feb 2014.
[94] FCC, "Measuring Mobile Broadband Methodology - Technical Summary," http://www.fcc.gov/measuring-broadband-america/mobile/technical-summary, Nov. 2013.
[95] J. Kastrenakes, "FCC releases Android speed test app to gather data on cell carrier performance," http://www.theverge.com/2013/11/14/5105090/fcc-launches-android-mobile-speed-test-app, Nov 2011.
[96] S. Silbert, "FCC launches speed test app for Android, looks to collect mobile broadband performance data," http://www.engadget.com/2013/11/14/fcc-launches-speed-test-app-android/, Nov 2011.
[97] Z. Honig, "FCC Speed Test app for iOS lets the government track your iPhone's network performance," http://www.engadget.com/2014/02/25/fcc-speed-test-app-ios/, Feb 2014.
[98] K. Bell, "FCC Launches iOS 'Speed Test' App," http://mashable.com/2014/02/25/fcc-speed-test-app-ios/, Feb 2014.
[99] FCC, "FCC Speed Test App Tip Sheet," https://www.fcc.gov/guides/mobile-speed-test-tip-sheet, 2014.
[100] S. Muckaden, "MySpeedTest: active and passive measurements of cellular data networks," Ph.D. dissertation, Georgia Institute of Technology, 2013.
[101] N. Feamster, "My Speed Test Mobile Performance Measurement Tool Released," http://noise-lab.net/2012/06/02/my-speed-test-mobile-performance-measurement-tool-released/, Jun. 2012.
[102] S. Muckaden, "MySpeedTest: Active and Passive Measurements of Cellular Data Network Performance," http://www.caida.org/workshops/isma/1302/slides/aims1302_smuckaden.pdf, Feb. 2013.
[103] Webpagetest, "Test a website's performance," http://www.webpagetest.org/, 2008.
[104] Guy Podjarny, "Open-Sourcing Mobitest," https://blogs.akamai.com/2012/03/open-sourcing-mobitest.html, Mar 2012.
[105] A. Dyke, "What is a HAR File and what do I use it for?" http://www.speedawarenessmonth.com/what-is-a-har-file-and-what-do-i-use-it-for/, Aug 2012.
[106] Akamai, "Test Your Website Performance On A Mobile Device," http://www.akamai.com/html/awe/login.html?campaign_id=F-MC-16282&curl=/html/awe_auth/mobitest.html, 2012.
[107] M. Piatek, "Measurement @ Google," http://www.caida.org/workshops/aims/1403/slides/aims1403_mpiatek.pdf, Mar. 2014.
[108] "Akamai Mobitest: Mobile Web Performance Measurement Agents," https://code.google.com/p/mobitest-agent/source/browse/trunk/mobitest-agent/Android/BZAgent/README?r=2, Mar 2012.
[109] N. Vallina-Rodriguez, A. Auçinas, M. Almeida, Y. Grunenberger, K. Papagiannaki, and J. Crowcroft, "RILAnalyzer: A Comprehensive 3G Monitor on Your Phone," in *ACM IMC*, ser. IMC '13. New York, NY, USA: ACM, Oct. 2013, pp. 257–264.
[110] A. Aucinas, N. Vallina-Rodriguez, Y. Grunenberger, V. Erramilli, K. Papagiannaki, J. Crowcroft, and D. Wetherall, "Staying online while mobile: The hidden costs," in *ACM CoNEXT*, ser. CoNEXT '13. New York, NY, USA: ACM, Dec. 2013, pp. 315–320.
[111] N. S. N. S. Labs, "Understanding smartphone behavior in the network," http://www.nokiasiemensnetworks.com/sites/default/files/document/Smart_Lab_WhitePaper_27012011_low-res.pdf, 2011.
[112] "NetNetwork," https://github.com/pragma-/networklog, 2011.
[113] S. Rosen, H. Luo, Q. A. Chen, Z. M. Mao, J. Hui, A. Drake, and K. La, "Discovering Fine-grained RRC State Dynamics and Performance Impacts in Cellular Networks," in *ACM Mobicom*, Sep. 2014.
[114] MobiPerf, "Data Collection and Privacy Policy," http://www.mobiperf.com/privacy, 2014.
[115] Y. Zhou, "Mobiperf," http://www.caida.org/workshops/isma/1302/slides/aims1302_yyzhou.pdf, Feb 2013.
[116] J. Rula, "ALICE - Technical Description," http://aqualab.cs.northwestern.edu/262-details-alice, Aug. 2014.
[117] Aqualab, "Namehelp," https://play.google.com/store/apps/details?id=edu.northwestern.aqualab.namehelp&hl=en, Apr 2014.
[118] ——, "Application Time (AppT)," https://play.google.com/store/apps/details?id=edu.northwestern.aqualab.behavior.research, Apr 2014.
[119] NUStudents, "NU Signals v2," https://play.google.com/store/apps/details?id=edu.northwestern.nux, May 2014.
[120] "Ookla," http://www.ookla.com/, 2015.
[121] Ookla, "Speedtest.net," https://play.google.com/store/apps/details?id=org.zwanoo.android.speedtest, Mar. 2015.
[122] ——, "Host a Speedtest Server," http://www.ookla.com/host, 2015.
[123] Ookla SpeedTest, "Mobile Test Server Selection," https://support.speedtest.net/hc/en-us/articles/203845480-Mobile-Test-Server-Selection, Oct. 2012.
[124] ——, "How do I correct my location?" https://support.speedtest.net/hc/en-us/articles/203845660-How-do-I-correct-my-location-, Oct. 2012.
[125] MaxMind, "GeoIP2: Industry Leading IP Intelligence," https://www.maxmind.com/en/geoip2-services-and-databases, 2012.
[126] Ookla SpeedTest, "How does the Begin Test button select a server?" https://support.speedtest.net/hc/en-us/articles/203845410-How-does-the-Begin-Test-button-select-a-server-, Jan. 2012.
[127] ——, "How does the test itself work? How is the result calculated?" https://support.speedtest.net/hc/en-us/articles/203845400-How-does-the-test-itself-work-How-is-the-result-calculated-, Jan. 2012.
[128] PingTest.net, "Measuring Network Quality," http://www.pingtest.net/learn.php, 2014.
[129] Ookla SpeedTest, "Ookla SpeedTest Mini," http://www.speedtest.net/mini.php, 2014.
[130] RadioOpt GmbH, "Traffic Monitor & 3G/4G Speed," http://www.trafficmonitor.mobi/en/download/, 2014.
[131] ——, "Download Traffic Monitor," http://www.trafficmonitor.mobi/en/download/, 2014.
[132] CacheFly, "Technology and Infrastructure," http://www.cachefly.com/cachefly-cdn/technology/, Mar. 2015.
[133] Google Developers, "Google Maps Developer Documentation," https://developers.google.com/maps/documentation/, Mar. 2015.
[134] OpenSignal.com, "OpenSignal WiFi map, speedtest," https://play.google.com/store/apps/details?id=com.staircase3.opensignal&hl=en, Mar. 2015.
[135] OpenSignal, Inc., "OpenSignal - Signal Finder and 3G/4G/Wifi Coverage Maps," https://itunes.apple.com/app/opensignal/id598298030, Mar. 2013.
[136] J. Cainey, B. Gill, S. Johnston, J. Robinson, and S. Westwood, "Modelling download throughput of LTE networks," in *Local Computer Networks Workshops (LCN Workshops), 2014 IEEE Conference on*, Oct. 2014.
[137] OpenSignal, Inc., "The State of LTE," http://opensignal.com/reports/state-of-lte/, Feb. 2013.
[138] FierceWireless, "3G/4G wireless network latency: Comparing Verizon, AT&T, Sprint and T-Mobile in February 2014," http://www.fiercewireless.com/special-reports/3g4g-wireless-network-latency-comparing-verizon-att-sprint-and-t-mobile-feb, Mar. 2014.
[139] Johanna, "OpenSignal Blog," http://opensignal.com/blog/2015/01/09/our-academic-partners/, Jan. 2015.
[140] OpenSignal Developers, "NetworkStats API," http://developer.opensignal.com/networkrank/, 2014.
[141] ——, "Tower Info API," http://developer.opensignal.com/towerinfo/, 2014.

[142] OpenSignal, Inc., "How phone batteries measure the weather," http://opensignal.com/reports/battery-temperature-weather/, Aug. 2013.
[143] ——, "OpenSignal Blog," http://opensignal.com/blog/2012/11/29/new-permissions-in-version-1-99-and-how-to-check-whether-an-app-is-malicious/, Nov. 2012.
[144] Vodafone, "Take control of your data and Wi-Fi usage – Vodafone NetPerform," http://www.vodafone.co.uk/discover-vodafone/apps-and-downloads/vodafone_netperform/, Mar. 2015.
[145] ——, "Vodafone Net Perform," https://play.google.com/store/apps/details?id=com.vodafone.netperform.full, Jun. 2014.
[146] ——, "Vodafone Net Perform," https://itunes.apple.com/ie/app/vodafone-net-perform/id946160163?mt=8, Jun. 2014.
[147] ——, "Vodafone Net Perform – Terms and Conditions," https://www.vodafone.co.nz/legal/terms-conditions/netperform/, Mar. 2015.
[148] SpeedSpot, "SpeedSpot: Pioneering Hotel WiFi Speed Test," http://speedspot.org/, Mar. 2015.
[149] Sensorly, "Unbiased Wireless Network Information. From people just like you." http://sensorly.com/, 2013.
[150] RootMetrics, "The RootMetrics testing methodology," http://www.rootmetrics.com/us/methodology, Mar. 2015.
[151] F. Kaup, F. Jomrich, and D. Hausheer, "Demonstration of NetworkCoverage – A Mobile Network Performance Measurement App," *IEEE NetSys*, March 2015.
[152] V-Speed, "Cloud Managed Speed Test," http://www.v-speed.eu/, Mar. 2015.
[153] NetRadar, "What's my mobile operator's coverage?" http://www.netradar.org/en, Mar. 2015.
[154] Cisco Systems, Inc., "Cisco Data Meter," http://ciscovni.com/data-meter/index.html, May 2013.
[155] Veloxity, Inc., "4Gmark Mobile performance test," http://www.4gmark.com/, Nov. 2014.
[156] nPerf.com, "Whats nPerf? How does it work?" http://www.nperf.com/en/, Nov. 2014.
[157] Internet2, "Network Diagnostic Tool (NDT)," http://software.internet2.edu/ndt/, 2013.
[158] N. Vallina-Rodriguez, N. Weaver, C. Kreibich, and V. Paxson, "Netalyzr for Android: Challenges and opportunities," in *Workshop on Active Internet Measurements (AIMS)*, Mar 2014.
[159] E. Gregori, L. Lenzini, V. Luconi, and A. Vecchio, "Sensing the Internet through crowdsourcing," in *Proceedings of the Second IEEE PerCom Workshop on the Impact of Human Mobility in Pervasive Systems and Applications (PerMoby)*, May 2013.
[160] "Portolan network tools," https://play.google.com/store/apps/details?id=it.unipi.iet.portolan.traceroute&hl=en, May 2014.