Using the Middle to Meddle with Mobile

Paper 148

ABSTRACT

Mobile networks are arguably the most popular, fastest growing systems in today's Internet ecosystem. However, these networks are also the least understood, largely due to locked-down operating systems running on devices that interact over a closed and opaque mobile network. This severely limits users' control over how their devices use mobile networks and restricts researchers' ability to deploy and test new in-network functionality at scale.

In this paper we present *Meddle*, a framework that enforces transparency in mobile networks and enables a new point of control for meddling with mobile traffic to improve user experience. We argue that *Meddle*, which builds upon VPNs and middleboxes, is feasible to implement, scalable and is capable of providing sufficient incentives for adoption by a large user base. In addition, *Meddle* provides a powerful framework that enables new research directions in mobile networks.

1. INTRODUCTION

Mobile networks are the most popular, fastest growing and least understood systems in today's Internet ecosystem. Despite a large collection of privacy, policy and performance issues in mobile networks [6, 10, 14, 22], users and researchers are faced with few options to characterize and address them. The crux of the problem is that mobile systems consist of walled gardens inside gated communities, i.e., locked-down operating systems running devices that interact over a closed and opaque mobile network.

From the user perspective, the problem is that subscribers to mobile networks have little control compared to what they have in their wired access networks. For example, in the home network, users are accustomed to the ability to install custom applications that change application network usage (e.g., ad blocking) and to run home routers that implement policies such as network access prioritization and parental controls. In the mobile environment, however, users are forced to interact with a single operating system tied to their device, generally use closed-source apps provided for the OS that routinely violate user privacy [10], and subscribe to network providers that can (and do) transparently modify, block or otherwise interfere with network traffic [25]. Users thus need a mechanism to take back control of how their devices use the mobile networks they pay for.

The lockdown of mobile devices places researchers in a similar bind. To characterize mobile traffic and design new protocols and services that are better tailored to the mobile environment, we would like a framework that allows us to intercept and potentially modify traffic generated by mobile devices as they move with users, regardless of the device, OS or carrier. However, implementing this functionality is difficult on mobile devices because it requires warranty-voiding techniques such as jail breaking to access and manipulate traffic at the network layer [6]. Even when using such an approach, carriers may manipulate traffic once it leaves the mobile device [25], thus rendering some research impractical. Last, some protocols and services should be implemented in the network instead of the device (e.g., prefetching and security filters) but researchers generally have no ability to deploy such solutions.

In this paper, we argue that we can provide the necessary framework to simultaneously address these issues for users and researchers by using middleboxes accessible through VPN tunnels, an approach we call *Meddle*. *Meddle* works for nearly all mobile devices out of the box: Android, iOS, and Blackberry all support VPNs, thus providing a portable mechanism to tunnel traffic to servers outside of the carriers' control regardless of the mobile device's network. Once packets arrive at VPN servers, we can use a variety of middlebox approaches to transform traffic to and from mobile devices. This enables new research in both measuring and characterizing mobile traffic, and designing new in-network features to improve the mobile experience.

In addition to avoiding transparent interference from any middleboxes inside mobile carrier networks, *Meddle* enables researchers to investigate what-if scenarios for the impact of new middleboxes as if they were deployed

in carrier networks. In § 3, we discuss several research directions enabled by this service including new app-accelerators, mobile-specific security filters and protocol manipulation to improve power consumption and data volume usage. Importantly, service providers and users can take advantage of these features without requiring any support from carriers or new OS-specific apps installed by users.

To obtain a large number of participating users for characterization and experimental evaluation, we need to explicitly align the goals of researchers and users. We argue that *Meddle* offers sufficient incentives for users to adopt the service by offering device-wide ad-blocking, privacy/security filters and parental controls at the network layer – functionality that mobile network providers do not currently make available.

While the technologies that enable *Meddle* are well understood, there are a variety of challenges and open questions we must address to ensure a solution that is practical both for end users and researchers. Importantly, we must design a VPN-server deployment that does more good than harm with respect to performance and power consumption so that we avoid a disincentive for usage (§4). In addition, we would like to investigate the extent to which we can enact optimizations that reduce page load times, improve security, reduce data consumption and potentially even conserve power – all from a middlebox that resides neither on the device nor in the carrier's network. Finally, we must address the additional security, trust and privacy concerns that arise when tunneling traffic outside of carrier networks into a third-party distributed service. We discuss these issues and others in Section 5.

2. DESIGN AND ARCHITECTURE

This section discusses the goals of our approach, details how alternative solutions do not address them and describes the architecture for *Meddle*.

2.1 Design Goals

Our goal is to provide an environment that facilitates characterization and experimentation for real mobile network traffic. To evaluate existing and new protocols and services in the mobile environment, we would like to have a continuous, fine-grained and representative view of network usage in the wild coupled with the ability to interpose on this traffic in real time. *Meddle* uses a VPN to connect users to a software middlebox; we briefly discuss two other options to achieve these goals and how they are infeasible.

One approach to achieving these goals is to work with mobile network providers. Unfortunately, these providers typically require an NDA to access traces of network usage (if they allow access at all), making it impractical to work with a large number of providers and thus biasing our view towards a small subset of mobile networks. Even with an NDA, implementing new functionality in the network can be challenging because carriers may not want to subject their paying subscribers to experimental protocols. Last, even with new functionality deployed in a carrier's network, there exist a large variety of other middleboxes already deployed in carrier networks that may block, modify or otherwise interfere with our ability to evaluate it.

Another approach is to use software running on or near mobile devices to capture and modify network traffic. For example, previous work has used access points or home routers to characterize home network traffic [18]. However, mobile devices typically travel distances greater than a single Wi-Fi AP, making a fixed hardware middlebox approach too limited for our purposes. One can also implement taps on network interfaces on the mobile devices themselves. Unfortunately, mobile apps cannot access raw network traffic without modifications to the mobile operating system, a process that can lead to warranty voiding and thus limits its practicality for broad deployment. Even with a rooted phone running a modified OS, collecting, storing and analyzing extensive network traces may consume an unacceptable amount of power and storage space.

2.2 Architecture

The key idea behind the *Meddle* architecture is to take two well-known technologies – VPNs and middleboxes – and combine them in unintended ways for the mobile environment. Specifically, major mobile OSes provide built-in VPN functionality for enterprise customers to enable access to resources in the enterprise's private network for employees "on the road". In *Meddle*, we use VPNs as a portable mechanism[1] to tunnel traffic from mobile devices to a machine outside of the carrier's network for the purpose of analysis and interposition. On *Meddle* servers, we use the StrongSwan open-source VPN implementation [23] that provides native IPsec functionality and runs on all modern Linux kernels. Middleboxes are traditionally used in managed networks (e.g., in enterprises and ISPs) to implement policies and enhanced services over IP. In *Meddle*, we use middleboxes as a mechanism not only to implement custom policies and services for users and service providers, but also for measuring networks and experimenting with alternative protocols for the mobile environment without requiring access to mobile carrier networks.

The architecture of *Meddle* is relatively straightforward. It consists of 3 components: *Meddle* servers, a redirector and profile storage (see Fig. 1). At a

[1]Andriod, BlackBerry and iOS all support VPNs natively, representing more than 86% of the mobile device market [7].

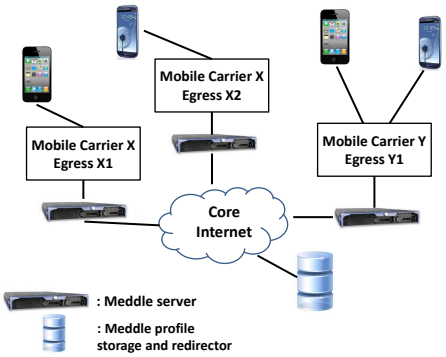

Figure 1: Meddle architecture, using two carriers (X and Y) and three egress points (X1 and X2 for carrier X and Y1 for carrier Y). Devices are dynamically mapped to *Meddle* servers near each egress point, and the profile manager ensures that device-specific middlebox settings migrate with users as they are mapped to different *Meddle* servers.

high level the redirector matches mobile clients with *Meddle* servers, the *Meddle* servers provide the VPN and middlebox services, and the profile storage manages device-specific policies that *Meddle* enacts.

When a device connects to the service, we direct it to a nearby *Meddle* server in a similar way to how the Akamai CDN uses DNS to redirect Web clients to nearby content caches [2]. In this case, the *Meddle* redirector sends VPN clients to a server that is relatively near the location at which data exits the mobile carrier's network to enter the Internet.

The mobile devices then establish a VPN tunnel to a *Meddle* server, which could run in a hosting center, on a distributed platform such as PlanetLab, EC2 or Compute Engine, or in a user's home network. Because the mapping between a device and its *Meddle* server may change over time and as users travel with their devices, we would like to be able to migrate the device-specific middlebox settings with them in a platform- and location-independent way. When the *Meddle* server authenticates the mobile device using the VPN's credentials, it downloads the middlebox settings (e.g., user-selected content filters and services) for the device from profile storage.

3. MEDDLE EXAMPLES

In this section, we discuss several new applications and research directions that *Meddle* enables in mobile systems. We separate these into two categories: (1) direct user incentives that will encourage broad adoption and thus provide a large subscriber base for experimentation and (2) research questions that can be answered using *Meddle* as a platform.

3.1 Direct User Incentives

To be useful for characterization and for evaluating new protocols and services over mobile networks, we would like to have a large, diverse user population that adopts *Meddle* and becomes available for testing experimental features. Similar to previous work in wired networks [3], we believe that direct incentives are essential for user adoption. We now list several incentives that we believe to be generally appealing for users. A common feature of the following examples is that they require raw access to network traffic, something not available by default on today's mobile OSes.

Privacy and Security. Enck et al. [6] show that many apps send data such as location and unique identifiers (IMEI) in the clear to third parties. We can use *Meddle* to search for known patterns of personally identifiable information and automatically strip or replace them with placeholders [10]. We can also allow users to observe how the installed apps use the network and with whom these apps share (or leak) information, a system similar to Mozilla Collusion [12]. In addition, a system like *Meddle* can host a service like the anonymization proxy described in Privad [9].

Ad blocking. In the desktop environment, users often install ad-blocking extensions to Web browsers to block content loaded by known ad servers.[2] This not only removes what are usually annoying images or animations, but also speeds Web page load time by reducing the number of objects fetched. In the mobile environment, only Firefox Mobile provides the ability to install ad-blocking extensions; however, there is no solution that applies to the myriad other apps that display ads. Using rules similar to tools like AdBlock, *Meddle* can simply block requests, rewrite Web pages or return zero-size objects for known ad servers. In this environment, the functionality will not only improve performance and privacy but can potentially save on power and data quota that otherwise would be wasted on fetching ad content.

Parental controls. Most mobile devices provide some form of parental controls, e.g., on iOS devices, parents can set which apps can be used, whether new apps can be installed and what type of content can be viewed. However, these controls do not provide the ability to filter based on network protocols or to limit network usage. Using *Meddle*, we can augment existing parental controls (which prevent modifying phone settings) by enforcing fine-grained bandwidth quotas and otherwise shape network usage based on time of day, protocols and sites being contacted.

3.2 Research Directions

Most research in mobile networks currently occurs at the edge (by installing new OSes or apps on devices) or behind closed doors of carrier networks. By exporting user traffic outside these closed networks, *Meddle* re-opens the network and makes it available for experi-

[2]For example, AdBlock[Plus] (ABP) has been installed by more than 20 million users.

mentation. In this section, we focus on a subset of interesting research directions *Meddle* enables. § 5 addresses the important ethical and privacy concerns when experimenting with human subjects.

Network usage characterization. Previous work that examined the network usage of mobile devices and apps is limited to lab studies or small campus deployments [6, 24]. With *Meddle*, researchers have the opportunity to collect network usage information from a large number of users worldwide without deploying any new hardware. Network traffic will enter *Meddle* regardless of where the user accesses the network, providing a continuous view of how mobile devices interact with the network. Further, *Meddle* provides an ideal vantage point for separating mobile-network performance from server-side performance, thus improving bottleneck identification for mobile applications.

Content coalescing, caching and prefetching. Several existing Web browsers for mobile devices [13, 20, 21] have explored alternative points in the design space for offloading the work of caching Web content, processing the DOM for display and prefetching pages to reduce latencies. While it seems clear that these approaches work in at least some real scenarios, it is unclear which approach is best when faced with typical user interactions. In fact, it is not clear *a priori* that any single approach for Web acceleration is optimal; e.g., perhaps the right combination of approaches depends on the site and on user behavior. With a large collection of traces gathered from *Meddle*, researchers have the unique opportunity to evaluate the effectiveness of these accelerators based on traffic "in the wild."

Mobile offloading. The mobile environment offers numerous opportunities for adopting a model with functionality "split" between mobile devices and wired, well-provisioned servers [4, 5]. Ideally, the servers would be placed inside carrier networks, thus reducing the round-trip times; in *Meddle* we can simulate this approach and play what-if scenarios by using *Meddle* servers.

We envision that most of the interesting opportunities for mobile offloading will come at the intersection of severe constraints for mobile devices (power, data volume quota and latencies) and the applications that extensively exercise those constraints. For instance, distributed hash tables (DHTs) are an example of a distributed service that has become critical for a variety of applications from P2P communication to content caching and anonymous networking. Due to the nature of the key-value store, each request for the value at a key entails a significant number of network operations. Using a split-application model, mobile device need only send a request for a key to the server-side and the server can perform all the network operations required to locate the value without consuming mobile network bandwidth.

4. FEASIBILITY

The previous section provided several examples of new research opportunities that *Meddle* enables. In this section, we highlight several key questions regarding the feasibility of deploying our approach at scale.

4.1 Overhead

Because our approach in part depends on users installing a VPN configuration and tunneling all traffic through *Meddle*, we evaluate whether the cost to the user in terms of performance, power and data quota is sufficiently low.

Power consumption. Tunneling traffic to a *Meddle* server requires that all traffic be encrypted by the mobile device. While this is already commonly performed for SSL connections, *Meddle* requires an *additional* layer of encryption. We observed a 10% increase in power consumption when streaming an HD video to Android and iPhone devices using our IPsec tunnel. We believe that this overhead is reasonably low, and we note that this cost for encryption comes with the added benefit of increased privacy from carriers.

An interesting research question is whether it is possible to *reduce* power consumption using *Meddle*. For example, Qian et al [15–17] found that traffic shaping (a service that *Meddle* provides) can significantly reduce the power consumed by devices when periodic application traffic and radio resource timers are out of sync.

Data consumption. IPsec encapsulation slightly inflates packet sizes, in addition to preventing carrier middleboxes from applying their own compression. We measured the overhead of the tunnel in terms of data overhead from IPsec headers and keepalive messages, finding that it ranges from 8–12%. For our measurements we setup *Meddle* as a VPN gateway for an iPhone and Android phone. On each phone we accessed the Internet using the VPN tunnel for about one hour. Our test traffic was generated by activities that we expect to be typical of mobile device usage: Web searches, map searches, online shopping, downloading popular apps, emailing and reading the news. We also uploaded a picture to Facebook and Twitter, streamed a video on YouTube, and played a popular game (Angry Birds).

Performance. By forcing user traffic to an intermediate server and interposing on flows, we may add latency both due to additional hops and due to processing time at the *Meddle* server. We envision a DONAR-style deployment where users are dynamically redirected to different *Meddle* servers based on network conditions and server load [26]. Given this model, we evaluate whether we can locate servers near mobile-network egress points using a deployment such as PlanetLab, and found that this is generally the case.

For this experiment, we used data from approximately

10 mobile phones located throughout the US and issued traceroutes from the devices to targets in Google and Facebook's networks. We then used the first non-private IP address seen from the mobile device on the path to a server. We assume that this corresponds to the first router adjacent to the mobile carrier's public Internet egress point. Note that we could not simply ping the device IPs because mobile carriers filter inbound ping requests. Using this set of egress adjacencies, we determined the round-trip time from each PlanetLab site, then took the average of the nearest five sites to represent the case where a host at the nearest site is unavailable due to load or other issues. The average latency to each router was between 3 ms and 13 ms, with a median of 5 ms. Thus, when compared to RTTs of 10s or 100s of milliseconds that exist in mobile networks, the additional latencies from traversing *Meddle* servers is expected to be relatively small or even negligible.

4.2 Deployability

In this section, we discuss the challenges for deploying *Meddle* at scale.

Portability. To be successful, *Meddle* should be supported by nearly all mobile devices and be easy to deploy on servers. In our current implementation, *Meddle* uses native IPsec to establish VPN tunnels and the Vyatta middlebox software to shape traffic. Both of these software artifacts are supported on vanilla Linux operating systems, which in turn run on nearly all servers. For mobile devices, Android, BlackBerry and iOS systems all support VPNs; Windows phones are expected to add support in version 8 of the OS. Manually installing a VPN generally requires filling out five fields on a Android phone and the VPN configuration can be distributed using a single file on iOS. We have tested that our server software runs correctly on Amazon's EC2 offering and are currently working on adding kernel modules to support it on PlanetLab and Vicci nodes.

Scalability. If wildly successful, we would like to ensure that *Meddle* scales gracefully and that there are sufficient resources to support large numbers of concurrent users worldwide. Based on our initial analysis using StrongSwan on commodity hardware, we found that each connection consumed on average less than 1% of CPU time. Thus, we expect to be able to support up to 100s or small number of thousands of users per server, which is in line with low-end VPN appliances sold by Cisco and Vyatta. A recent study [1] showed that current rates for 3G networks in the US were between 0.59 and 3.84 Mbps; assuming devices are uniformly distributed across carriers, we expect to be able to support 250 users (saturating their download capacity) for every 1 Gbps of bandwidth at the server.

5. DISCUSSION

In this section, we discuss several open questions, limitations of our approach and how to address them.

User incentives. We claim that users would like more control over how their mobile devices interact with the network and that the incentives for *Meddle* adoption are sufficient to attract a diverse and large user population. One could argue that users do not need or want more control over the network, and that non-experts would find it hard to take advantage of the opportunities that *Meddle* affords. In response to this objection, we note that large numbers of users and policymakers were outraged by recent studies about information leaks from apps [24]. Further, there is a large set of users installing ad-blocking software in Web browsers – if tens of millions of people do it, it likely falls in the domain of non-experts.

We do not expect any single incentive for user adoption to be sufficient. Rather, the research enabled by *Meddle* should form a positive feedback loop in which new, proven research artifacts become additional incentives for user adoption, thus enabling further research.

Trust and Privacy. *Meddle* eliminates the need to trust carriers or devices with network traffic; however, this comes at the cost of users needing to trust *Meddle* servers. We currently use state-of-the-art IPsec implementation where a user and a *Meddle* system authenticate themselves to each other using digital certificates.

Beyond the trust concerns, *Meddle* provides a tap on network traffic that could allow researchers to see all unencrypted packets generated by devices – a serious risk for violating user privacy. We must ensure that researchers are restricted to capturing only summary information about packets (e.g., headers and packet length), that user identifiers are removed from any persistently stored dataset and that users explicitly give informed consent for any tracing, experimental features or other forms of traffic manipulation. To help instill confidence from users, *Meddle* software will be implemented as open source artifacts made publicly available, and users will have the option to run their own instance of *Meddle* (with their own root of trust) if they so desire.

Limitations. While *Meddle* offers several new opportunities for research, there are limitations to the control it offers. First, *Meddle* currently cannot control what apps are installed on phones or the network traffic they generate and thus cannot address many network problems that occur "at the source." Second, although *Meddle* can modify packet timings to play nicely with protocols and timers in the mobile environment, it cannot be used to unilaterally effect new protocols between the mobile device and *Meddle* server. Last, *Meddle* has limited ability to impose DPI-based policies and protocols when a connection to a server is encrypted (e.g., via SSL). It is possible that users can install a root

of trust enabling *Meddle* servers to proxy these sessions, but this raises additional privacy concerns.

Interference. One of the key features of *Meddle* is that it takes control of network traffic away from mobile network providers. This is a half truth, because carriers can choose to block communication with *Meddle* servers. However, there are strong disincentives for this behavior because it not only violates net neutrality, but also is likely to pose a public relations nightmare for a carrier accused of doing so. In a similar vein, advertisers and the app providers that they support are likely to find ways to avoid our filters for blocking ads, much like the ongoing cat-and-mouse game in desktop Web browsers. We do not claim to have a solution to end the game, but we argue that the continued effectiveness of ad blocking in desktop scenarios is likely to persist in the mobile environment.

6. RELATED WORK

Meddle builds upon two existing technologies: VPNs and middleboxes. In our current implementation, we rely on an IPsec [11] implementation to tunnel traffics to *Meddle* servers. Sherry et al. [19] explore the opportunities enabled by moving middleboxes to the cloud, which includes simplifying management for enterprise network administrators. In contrast, our work focuses on the mobile capabilities software middleboxes enable rather than a redirection architecture for enterprise networks.

Meddle provides researchers with a cross-platform, cross-carrier technique for capturing network usage patterns from mobile devices. Previous work used on-demand active measurements to characterize network measurements [22,25]; however, these measurements are restricted to the point at which users run the tests. Gerber et al. [8] use passive measurements alone to estimate transfer rates in a single carrier's network; *Meddle* will enable such analysis across multiple carriers.

We expect to use *Meddle* to investigate security and privacy issues in the traffic that mobile devices generate. By monitoring and controlling information at lower layers in the software stack, previous work [6, 10, 24] has shown that existing apps leak significant private information. In *Meddle*, we take this downward mobility to an extreme, moving off the end-host entirely and eliminating the restrictions of a lab setting.

The CloneCloud [4] and MAUI [5] projects explored the space of moving functionality from mobile devices into the cloud. By interposing on user traffic, *Meddle* will allow us to explore some of this functionality without requiring device modifications.

7. CONCLUSION

We described *Meddle*, a platform for users to regain control of their mobile network traffic while giving researchers a foothold to deploy new mobile network services and to characterize usage behaviors in the wild. We showed that the overheads for *Meddle* are low and that *Meddle* can easily scale to thousands of users. We are currently using our prototype *Meddle* architecture to build out a system for public deployment.

8. REFERENCES

[1] 3G/4G performance map: Data speeds for AT&T, Sprint, T-Mobile, and Verizon. www.pcworld.com/article/254888/3g4g_performance_map_data_speeds_for_atandt_sprint_tmobile_and_verizon.html.

[2] AKAMAI. Akamai CDN. www.akamai.com.

[3] CHOFFNES, D. R., AND BUSTAMANTE, F. E. Taming the torrent: A practical approach to reducing cross-ISP traffic in peer-to-peer systems. In *Proc. of ACM SIGCOMM* (2008).

[4] CHUN, B.-G., IHM, S., MANIATIS, P., NAIK, M., AND PATTI, A. Clonecloud: elastic execution between mobile device and cloud. In *Proc. of Eurosys* (2011).

[5] CUERVO, E., BALASUBRAMANIAN, A., CHO, D.-K., WOLMAN, A., SAROIU, S., CHANDRA, R., AND BAHL, P. Maui: making smartphones last longer with code offload. In *Proc. of MobiSys* (2010).

[6] ENCK, W., GILBERT, P., CHUN, B.-G., COX, L. P., JUNG, J., MCDANIEL, P., AND SHETH, A. N. Taintdroid: an information-flow tracking system for realtime privacy monitoring on smartphones. In *Proc. of USENIX OSDI* (2010).

[7] Gartner smart phone marketshare 2012 Q1. www.gartner.com/it/page.jsp?id=2017015.

[8] GERBER, A., PANG, J., SPATSCHECK, O., AND VENKATARAMAN, S. Speed testing without speed tests: estimating achievable download speed from passive measurements. In *Proc. of IMC* (2010).

[9] GUHA, S., CHENG, B., AND FRANCIS, P. Privad: practical privacy in online advertising. In *Proc. of USENIX NSDI* (Berkeley, CA, USA, 2011), NSDI'11, USENIX Association, pp. 13–13.

[10] HORNYACK, P., HAN, S., JUNG, J., SCHECHTER, S., AND WETHERALL, D. These aren't the droids you're looking for: retrofitting android to protect data from imperious applications. In *Proc. of CCS* (2011).

[11] KENT, S., AND SEO, K. Security architecture for the internet protocol, 2008.

[12] Mozilla collusion. www.mozilla.org/en-US/collusion/.

[13] Opera mini browser. www.opera.com/mobile/features/.

[14] PATHAK, A., HU, Y. C., AND ZHANG, M. Where is the energy spent inside my app?: fine grained energy accounting on smartphones with eprof. In *Proc. of Eurosys* (2012).

[15] QIAN, F., WANG, Z., GAO, Y., HUANG, J., GERBER, A., MAO, Z., SEN, S., AND SPATSCHECK, O. Periodic transfers in mobile applications: network-wide origin, impact, and optimization. In *Proc. of WWW* (2012).

[16] QIAN, F., WANG, Z., GERBER, A., MAO, Z., SEN, S., AND SPATSCHECK, O. Profiling resource usage for mobile applications: a cross-layer approach. In *Proc. of MobiSys* (2011).

[17] QIAN, F., WANG, Z., GERBER, A., MAO, Z. M., SEN, S., AND SPATSCHECK, O. Characterizing radio resource aloptlocation for 3G networks. In *Proc. of IMC* (2010).

[18] Samknows & Ofcom UK broadband performance testing. www.samknows.com/broadband/ofcom_and_samknows, June 2009.

[19] SHERRY, J., HASAN, S., SCOTT, C., KRISHNAMURTHY, A., RATNASAMY, S., AND SEKAR, V. Making middleboxes someone else's problem: Network processing as a cloud services. In *Proc. of ACM SIGCOMM* (2012).

[20] Amazon silk browser. www.amazon.com/gp/help/customer/display.html?nodeId=200775440.

[21] SPDY: An experimental protocol for a faster web. www.chromium.org/spdy/spdy-whitepaper.

[22] Speedtest.net mobile. www.speedtest.net/mobile.php/.

[23] Strongswan. www.strongswan.org.

[24] THURM, S., AND KANE, Y. I. Your apps are watching you. accessed july 19, 2012. online.wsj.com/article/SB10001424052748704694004576020083703574602.html.

[25] WANG, Z., QIAN, Z., XU, Q., MAO, Z., AND ZHANG, M. An untold story of middleboxes in cellular networks. In *Proc. of ACM SIGCOMM* (2011).

[26] WENDELL, P., JIANG, J. W., FREEDMAN, M. J., AND REXFORD, J. Donar: decentralized server selection for cloud services. In *Proc. of ACM SIGCOMM* (2010).