Meddle: Traffic Indirection for Practical Mobile Diagnosis

Ashwin Rao[a], Amy Tang[b], Shen Wang[c], Nick Martindell[c], Justine Sherry[b], Arnaud Legout[a], Walid Dabbous[a], Arvind Krishnamurthy[c], and David Choffnes[d]
[a]INRIA. [b]University of California, Berkeley.
[c]University of Washington. [d]Northeastern University.

Abstract

We present, Meddle, a platform that relies on traffic indirection to diagnose mobile Internet traffic. Meddle is motivated by the absence of built-in support from ISPs and mobile OSes to freely monitor and control mobile Internet traffic; the restrictions imposed by mobile OSes and ISPs also make existing approaches impractical. Meddle overcomes these hurdles by relying on the native support for traffic indirection by mobile OSes. Specifically, Meddle proxies mobile Internet traffic through a software defined middleboxes configured for mobile traffic diagnosis. In this paper, we use Meddle to tests the limits of the network perspective of mobile Internet traffic offered by traffic indirection.

We use this perspective to characterize and control the behavior of mobile applications and provide a first look at ISP interference on mobile Internet traffic. [**TBD: We report on controlled experiments we conducted to analyze the network behavior of 100 most popular iOS and Android applications.**] [**TBD: We also report on how ISP interfere with mobile HTTP Internet traffic.**]

1 Introduction

Mobile systems consist of walled gardens inside gated communities, i.e., locked-down operating systems running on devices that interact over a closed and opaque mobile network. Despite a large collection of privacy, policy and performance issues in mobile networks [4, 6, **?**, 1, 7, 2], researchers are faced with few options to characterize and address them.

The key challenge is that mobile OSes and ISPs provide no built-in service to diagnose the network traffic generated by these applications. [**TBD: Why is this important? Why do we care?**]. As a result, previous studies [**?**, 5, 3, 4, 10, **?**] are constrained by at least one of the following: mobile OSes, access technology, device manufacturer, installed applications, and user behavior. In this work, we are the first to present an approach that compromises none of these, an approach that can be used across carriers, mobile devices, apps, and access technologies.

Meddle redirects all Internet traffic through a software-defined middlebox for the purpose of analysis and interposition. Specifically, *Meddle* builds on the native support for VPN tunnels and HTTP proxies by mobile OSes to tunnel all the Internet traffic regardless of the access technology used by the device. A shortcoming of this approach is that it compromises the fine grained view offered by existing solutions in favor of being user-friendly.

In this paper, we use *Meddle* to test the limits to which traffic redirection can be used to diagnose mobile Internet traffic. [**TBD: too weak also mobile Internet traffic needs to be rephrased.**]. The main contributions of this paper are as follows:

1. Platform for mobile diagnosis through network traffic analysis and control. Single server solution empowers users to install and configure them on home-gateways. Researchers can deploy them for measurement studies.
2. Controlled experiments using off-the-shelf Android and iOS devices.
3. Controlled experiments to analyze ISP interference in US and France.

The remainder of this paper is structured as follows.

2 Platform Description

In this section, we present an overview of *Meddle*.[1] The goal of *Meddle* is straightforward: *we seek to monitor and control a mobile device's Internet traffic with the aim to diagnose the behavior of mobile apps, OS services, and traffic interference from ISPs*. While previous

[1] *Meddle* is currently in private beta with deployments in the US, France and China. We will make the *Meddle* software publicly available.

work has accomplished this for a limited set of devices or networks, we seek to avoid such limitations. Indeed, there exists a trade-off between a practical user-friendly solution and a solution that offers a fine-grained control over mobile devices and the traffic they generate. Unlike existing approaches, we take the direction for a practical and user-friendly solution and test the limits of its usefulness. Specifically, we relinquish OS-level controls to focus on the Internet traffic generated by mobile devices and try to use this perspective to diagnose mobile apps, OS services, and the ISPs that serve these devices.

1. *Agnostic to OS, ISP, access technology, and applications.*. The research work that comes out must not be limited to specific OS, ISP, or device manufacturer, or set of applications.
2. *Always On.* Once enabled by the user, it must be passive and pervasive. It must not demand periodic inputs from end-users. Monitor traffic continuously, even when devices switch between networks or return from being idle.
3. *On-demand.* The user must be able to enable and disable service easily. It must be easy to fallback to the original state. This is important to ensure that users can easily opt-out and are not blocked if there are some problems with the system.
4. *Deployable.* Easy to install/use/configure. It must not require warrant voiding of the device. This is important to support a large user-base. *Meddle* should be equally feasible to deploy on a single machine or using a collection of VMs in a hosted/cloud deployment.
5. *Scalable.* Can support many users. To ensure that research results have statistical significance.
6. *Traffic Interposition.* We want to not only monitor, but also possibly modify data packets in order to experiment. This makes *Meddle* both a passive monitoring and an experimental platform.
7. *Encryption agnostic.* Achieve visibility of both encrypted and plaintext traffic.

This network perspective offered by traffic redirection is promising because mobile devices are increasingly becoming the primary gateway to access Internet based services [8]. This perspective becomes even more important because a large number of free applications use Internet connectivity with the sole purpose of serving advertisements that make-up for their costs [7, 9].

In the following, we describe in detail the design of *Meddle*, then we discuss its feasibility and limitations.

2.1 Design

To meet the above goals, we exploit the observation that nearly all devices support network traffic indirection via virtual private networks (VPNs) and HTTP proxies.

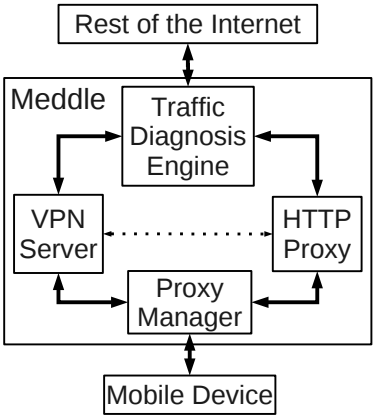

Figure 1: Figure showing traffic redirection and flow of traffic through VPN and HTTP proxy

HTTP proxies exchange traffic over the clear, thus offering the potential to identify ISP interference. The encryption offered by VPNs protects the device's network traffic from potential ISP interference. VPNs can thus be used to diagnose the traffic naturally generated by the mobile apps and OS services.

2.1.1 Architecture

Architecture as shown in Figure 1.

Explain two Proxies, one VPN and other HTTP.

Explain how each goal is met and why we need two proxies.

The role of the VPNs Proxy. VPNs allow tunnel everything (all IPv4 traffic). All IPv4 traffic can be monitored and controlled on the middlebox.

The role of HTTP Proxy. Check ISP interference. What How tripwires are implemented.

Though deployment of proxies appears to be straighforward they come with challenges that need to be addressed to make the solution practical and feasible. We now describe how we addressed these challenges.

2.1.2 Configuration requirements.

VPNs on mobile devices do not inherently tunnel all the Internet traffic. However, mobile devices support features that can be built on to enable tunneling of all Internet traffic.

All iOS devices (version 3.0 and above) support *VPN On-Demand*, which forces traffic for a specified set of domains to use VPN tunnels. To ensure all possible destinations match this list, we exploit the fact that iOS uses suffix matching to determine which connections should be tunneled; accordingly, we specified the domain list as the set of alphanumeric characters (a-z, 0-9, one character per domain). Android version 4.2 and above supports

an *Always On VPN* connection that provides the same functionality; for Android version 4.0 and above there is an app API that allows apps to manage VPN tunnels. We support both options. Both iOS and Android support VPN connections using the IPSec standard, This means we can implement our VPN proxy server using free and open-source code from Strongswan [**?**].

[**TBD: Tripwire**]

2.2 Existing Deployment

Where is this currently deployed?

What is the objective of this deployment?

Who are the ones signed up?

What was the incentive given?

2.3 VPN Overheads

Increase in Network Latency. We measured 50 VPN-connection establishment times on both iOS (iPhone 5 / iOS 6.1) and Android (Galaxy Nexus / Android 4.2), for Wi-Fi and cellular connections. The *Meddle* server was running on a university network. For Android (using IKEv2), the maximum establishment time was 0.81 seconds on Wi-Fi and 1.59 seconds on cellular. For iOS (using IKEv1), the connection takes longer due to the older protocol version: we observe a maximum of 2 seconds on Wi-Fi and 2.18 seconds on cellular. Because each VPN session supports many flows, the amortized cost of connecting is small. [**TBD: Cite results from latency to home gateways based on DSL results in PAM and IMC.**] [**TBD: Address comments in conext review on latency**]

Increase in Power Consumption. Mobile devices expend additional power to establish, maintain and encrypt data for a VPN tunnel. To evaluate the impact on battery, we used a power meter to measure the draw from a Galaxy Nexus running Android 4.2. We run 10-minute experiments with and without the VPN enabled. For each experiment, we used an activity script that included Web and map searches, Facebook interaction, e-mail and video streaming. The VPN leads to a 10% power overhead. [**TBD: min, max, median for results given by dave**]. For iOS devices, we relied on the battery readings provided by iOS because we cannot attach a power meter directly to the battery. We again found an approximately 10% power overhead of using VPNs when we drained a fully charged battery while performing the operations performed during the tests for Android devices.

Increase in Traffic Volume. *Meddle* relies on IPsec for datagram encryption, thus there is an encapsulation overhead for each tunneled packet. To evaluate this overhead, we use 30 days of data from 25 devices that to compare encapsulated and raw packet sizes. We observe a maximum encapsulation overhead of 12.8% (average approximately 10%). Within the scope of the traffic monitoring experiments performed with *Meddle*, the impact of this overhead is negligible. However, in case of experiments with a limited cellular data plan, this overhead must be taken into account

2.4 Limitations

At most one tunnel. [**TBD: Not true for http proxy**] Currently iOS and Android support exactly one VPN connection at a time. This allows *Meddle* to measure traffic over either Wi-Fi or cellular interfaces, but not both at once. The vast majority of traffic uses only one of these interfaces, and that interface uses the VPN

Proxy location. When traffic traverses the *Meddle* box, destinations will see the *Meddle* box address, not the device IP, as the source. This might impact services that customize (or block access to) content according to IP address (e.g., in case of localization). A solution to this problem is to use a *Meddle* instance with an appropriate IP address

ISP support. Some ISPs block VPN traffic, which prevents access to our current *Meddle* implementation. We note that few ISPs block VPN traffic, and there is an incentive not to block VPN traffic to support enterprise clients.

IPv6. *Meddle* cannot be currently used on networks using IPv6 because IPv6 is not fully supported by mobile devices. Indeed, we observe that though iOS and Android support IPv6 they currently do not support IPv6 traffic through VPN tunnels

MPTCP?.

2.5 Costs

Deployment and Running Costs

Trust Provider

2.6 Incentive for End-user Deployment

Deploy on Home Gateway. This is why we need the single machine constraint.

Packet Filtering. Custom ad blocks. Protect against data leaks.

Security from untrusted Wi-Fi APs.

Modular Architecture for Offloading Activities.

3 Applications and Malware

4 ISP Interference

5 Conclusion

References

[1] eDoctor: Automatically Diagnosing Abnormal Battery Drain Issues on Smartphones. In *Proc. of USENIX NSDI* (2013), pp. 57–70.

[2] BICKFORD, J., LAGAR-CAVILLA, H. A., VARSHAVSKY, A., GANAPATHY, V., AND IFTODE, L. Security versus Energy Tradeoffs in Host-Based Mobile Malware Detection. In *Proc. of MobiSys* (2011).

[3] CHEN, X., JIN, R., SUH, K., WANG, B., AND WEI, W. Network Performance of Smart Mobile Handhelds in a University Campus WiFi Network. In *Proc. of the Internet Measurement Conference (IMC)* (2012), pp. 315–328.

[4] 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 the USENIX Operating Systems Design and Implementation (OSDI)* (2010), pp. 1–6.

[5] GERBER, A., PANG, J., SPATSCHECK, O., AND VENKATARAMAN, S. Speed Testing without Speed Tests: Estimating Achievable Download Speed from Passive Measurements. In *Proc. of the Internet Measurement Conference (IMC)* (2010), pp. 424–430.

[6] HORNYACK, P., HAN, S., JUNG, J., SCHECHTER, S., AND WETHERALL, D. "These Arent the Droids Youre Looking For": Retrofitting Android to Protect Data from Imperious Applications. In *Proc. of CCS* (2011), pp. 639–652.

[7] 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).

[8] SANOU, B. ICT Facts and Figures. Tech. rep., International Telecommunications Union, 2013.

[9] VALLINA-RODRIGUEZ, N., SHAH, J., FINAMORE, A., GRUNENBERGER, Y., PAPAGIANNAKI, K., HADDADI, H., AND CROWCROFT, J. Breaking for Commercials: Characterizing Mobile Advertising. In *Proc. of the Internet Measurement Conference (IMC)* (2012), pp. 343–356.

[10] WANG, Z., QIAN, Z., XU, Q., MAO, Z., AND ZHANG, M. An Untold Story of Middleboxes in Cellular Networks. In *Proc. of the ACM SIGCOMM Conference* (2011), pp. 374–385.