Computer Network
Yingcong Li, S1616245
On the Accuracy of Smartphone-based Mobile Network Measurement

---

Nowadays, most of the mobile apps provide their services by connecting the network, so measuring and understanding the quality of network become more important to end-users and services providers. Although lots of network measurement apps can be downloaded from app markets in a different operating system, their measurement accuracy has not been proved and little known by end-users. Li et al. performed an accuracy measurement to these apps on Android platform and use Round-trip Time (RTT) as the metric.

To measure the delay, *delay overhead* (denoted as Δd) is used to shows the difference of RTT captured by the application layer (d_u) and the hardware layer (d_n).

$$\Delta d = d_u - d_n = (t_u^i - t_u^o) - (t_n^i - t_n^o).$$

Testbed consists of a measurement server and a wireless AP. Three Android phones are used--Google Nexus 5, HTC One, and Sony Xperia J to increase the diversity. They developed a test app for each of the main approaches which smartphone-based network measurement app used--Native ping, Inet ping and HTTP ping. *tcpdump* is also running in each of the phones to capture the time message arriving the kernel. Although the actual value d_n is not supported to be computed in the phone, they overcame this problem by deploying a multi-sniffers to help researchers recover timestamps for a missing frame, based on the timestamps captured in these 3 sniffers.

As the result, the RTT measured by apps are inflated significantly among three phones. The delay overheads range from less than 1 ms to over 16 ms. Delay for Nexus 5 is increasing along with the server delay increasing, while the other two phones keep a relatively more constant delay. Also, delay deflation can be observed in all three phone, reason for that is due to the coarse resolution of ping. On the other hand, the same Android phone also experiences different and significant delay for different measurement methods. To sum up, HTTP ping and Native ping exhibit smaller delay overheads for most of the case.

This delay is chiefly caused by the message passing time from kernel to hardware (denoted as Δd_k). But they failed to figure out the reason for Δd_k or the source of this delay (in Hardware or Kernel?). Evidence shows that message passing time from application to the kernel (denoted as Δd_u) is very small (very close to 0) in Native ping, Δd_u in Inet ping is also small, except of HTTP ping which is much larger (around 5.5ms to 7.5 ms). And these delay overheads are asymmetric between sending and receiving sides. Delay and asymmetry in this part is are caused by the runtime environment of a app, whereas the app in the in-app approach runs as an instance of DVM, so more instructions have to be executed than the native Linux program. By bypassing the DVM using simple C socket program, the Δd_u can be greatly mitigated.

Finally, pre-compiled C program which supports RTT measurements with HTTP messages is recommended. The delay overhead introduced in application-kernel can be mitigated to ~1 ms in both Inet ping and HTTP ping. And total delay overheads can be controlled under 5 ms. This methodology can be widely deployed in the apps since it does not need the root privilege. Although HTTP ping will introduce 0.4ms more delay because HTTP needs further processed in application, but handling TCP SYN/RST packets is complete in kernel. In the future, experiments can be conducted to iOS and Windows phone as well, or extending to cellular network. What's more, the cause of the delay incurred in Δd_k is worth exploring.