Application for the emulation of PingER on Android Devices

Aditya Pan
Department of CSE, ASET
Amity University
Noida, UP, India
pan.aditya93@gmail.com

Prof. (Dr.) Abhay Bansal
Department of CSE, ASET
Amity University
Noida, UP, India
abansal1@amity.edu

Prof. (Dr.) Bebo White
SLAC National Accelerator Laboratory
Stanford, CA, USA
bebo@slac.stanford.edu

***Abstract*— PingER is an end to end internet performance measurement tool. It was originally developed by the Stanford Linear Accelerator Centre as an IEPM tool. It is used as a tool to test the Digital Divide from an internet performance viewpoint. PingER measurements are done by measurement agents (MA) in approximately 60 sites in around 20 countries. A number of hosts are setup globally for target measurement purposes, and there are roughly 700 sites in over 160 countries dedicated to the same. However using a desktop machine for the purpose causes a massive power drain of around 100W an hour per machine. This paper discusses the implementation of an Android based mobile application to be used as a pingER monitoring agent, through which the total space and energy requirements can be minimized, while leveraging the advantages of Android smartphones.**

Keywords*— Android; Network Monitoring; Ping; PingER; IEMP Tool

I. INTRODUCTION

A PingER is an IEMP (Internet End to End Performance Measurement) tool developed by Stanford Accelarated Reasearch Centre (SLAC). It stands for Ping End-to-End Reporting and was started to identify the Digital Divide in terms of internet performance. Digital Divide [1] is a condition of economic or social inequality with reference to the access, usage and impact of ICT (Internet Communication Technologies. PingER provides data for the study of the Digital Divide on an international scale and through the use of roughly 700 sites in over 100 countries.

PingER measurements are made by aproximately 60 MAs(Measuring Agents). They make measurements to over 700 targets in 160 countires, i.e to an estimated 99% of the world's connected population. Measuring agents have to be online 24 x 7 x 365 hours a year. This leads to high usage costs. The average desktop computer has a wattage of 80-250, depending on non-essential peripherals and components installed. The cost to operate a typical computer and monitor system of 130 watts keeping in mind the uptime requirements of an MA is $129.73 [12]. The operating cost of a smartphone is much lesser and is around $1.36 a year [13]. Moreover a desktop and peripherals has considerable space requiremnts when compared to the small and compact size of a smartphone. This paper proposes an Android application which will attempt to provide functionality similar to a PingER MA, which is targeted for the Android Lollipop (5.x) with and has been compiled using SDK API 22 [16].

The rest of the paper is organized as follows: Section II discusses the essential workings of PingER project and Android system pertaining to the paper requirements. Section III discusses the methodology; the sequence of actions followed by the prototype and all assumptions that have been made, and concludes by displaying the prototype. Section IV presents the outputs and the data obtained through the use of the application. Section V concludes the paper.

II. BACKGROUND

A. PingER Process

PingER uses measurement agents which will periodically ping target sites and are responsible for the storage and analysis of the received data. The data can be used for deriving various metrics, which can be useful for the purposes of determining throughput, Voice over Internet Protocol (VoIP), streaming, haptics and more. Another interesting field this can be used in is in the determination of the geolocation of a host by sending pings to it from well-known landmarks [3].

1) Measurement Process of a MA

PingER uses the ping program [18] to send ICMP (Internet Control Messages Protocol) packets to a remote target and processes the response or unresponsiveness of the target.

PingER measurements are made by aproximately 60 MAs(Measuring Agents). They make measurements to over 700 targets in 160 countires, i.e to an estimated 99% of the world's connected population. The measurement is done in cycles of roughly 30 minute intervals. In each cycle, a MA will issue a 100 byte and a 1000 byte ping request to each target host in the list belonging to the concerned MA(7). The MA will stop sending ping requests when either following condition holds true:-

a) The MA has received 10 ping responses.

b) The MA has sent 30 ping requests.

Therefore Ping responses are recorded as N, where 0<=N<=10

2) Measurement Metrics

The received data is recorded and archived, where a particular entry exists for each set of pings from a MA to a particular target. The recorded data for any ping set consists of:

the MA name and its IP address; the target names and IP addresses; a time-stamp; the number of bytes in the ping request; the number of ping requests and responses (N); the minimum Round Trip Time (RTT) (Min_RTT), the average RTT (Avg_RTT) and maximum RTT (Max_RTT) of the N ping responses; followed by the N ping sequence numbers, followed by the N RTTs [6].

From the N RTTs we derive various metrics - including: minimum ping RTT; average RTT; maximum RTT; loss; and reachability (host is unreachable in case of 100 % packet loss); standard deviation (stdev) of RTTs, 25% probability (first quartile) of RTT; 75% probability (third quartile) of RTT; Inter Quartile Range (IQR). We also derive the Inter Packet delay (IPD) and the Inter Packet Delay Variability (IPDV) as the IQR of the IPDs [5][7].

B. Android System

Android is currently the largest smartphone operating system in the world with an estimated 79 % market share [15]. The system is based on Linux but has been heavily modified to overcome the limitations of mobile devices including limited battery and screen size. This paper is mainly concerned with certain components of the system;

1) Android Kernel

The Android Kernel directly uses one of the branches of the Linux Long Term Support (LTS). As of April 2014, 3.4 or 3.10 are mostly used versions of the Linux Kernel in use The Android Kernel however has many architectural changes outside the typical development cycle of Linux. Google's variant of the kernel has several architectural changes, such as out-of-memory handling (OOM), logger, wakelocks and more [10].

Many middleware, libraries and APIs written in C, along with application framework along with Apache Harmony based Java-compatible libraries running application software exist on top of the kernel.

2) Features of various Android Versions

- Dalvik

Pre-Lollipop (version 5.x) versions of Android had Dalvik as a process virtual machine, along with trace-based JIT (Just In Time Compilation), which was used to execute dex-code translated from bytecode. Additionally, Dalvik had to compile and natively executed of frequently used code selectively, called "traces" upon every application launch instance [11].

- Android Runtime (ART)

Launched originally in Android version 4.4 (Kitkat) and used by default since Lollipop, this new environment was introduced by Android, which directly converts bytecode to machine code upon application installation or AOT (Ahead of Time compilation).

3) Security in Android

By default an application cannot perform operations that will significantly impact the operating system, user or other applications. This pertains to user data such as contacts, emails or even application files. Each application is executed in its own process sandbox. Some basic features are provided by each sandbox, however additional features require explicitly stating so in the application manifest, and the user is prompted for the same during installation. The "permissions" needed have to be explicitly declared and Android system requests the user for consent.

4) Rooting

In Android it is possible to get in-application access to the Linux kernel, provided the concerned phone has been rooted. Rooting allows us to bypass Android security protocols [17] which will prevent the application from working, since ping requests will be blocked. Rooting poses a security risk in terms of integrity and privacy of user data, but is justified on the grounds that PingER provides full public access to all data and metrics and the phone can be setup with dummy accounts.

III. METHODOLOGY AND EXPERIMENTAL SETUP

The application is designed to behave as a PingER Measuring Agent to meet the need of lesser power consumption and space consumption.

A. Model Process Flow

The process flow has been kept very similar to the logic of the pingER software, and is summarised in a flowchart as shown in figure 1.

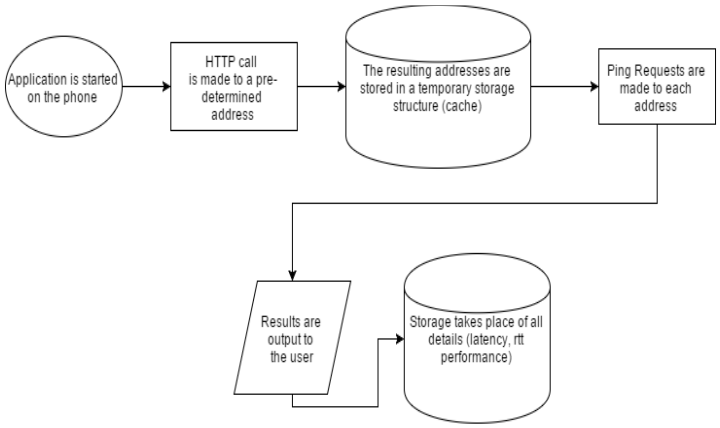

Figure 1: Process flow of Model

B. Assumptions and Simplifications

Certain assumptions have been made to ensure the proper functioning of the application prototype, and also certain simplifications have been made to make the process work without exceptions or failures being explicitly handled.

a) Device is connected to the internet using a reliable and stable connection.

b) Device has charge or is connected to the charger.

c) The device is running Android 4.0 or above.

d) The phone is rooted, either through special software or through custom ROMs.

e) The application has been granted root acces.

f) Neither the internet connection nor the power ever fails.

g) The number of target sites has been reduced to 7 but can be scaled up.

C. *Environment Setup*

For the purpose of the paper, certain tools have been used to create, design and test the application. The following is the list of tools used for the purpose

a) Android Studio 1.4 with compiler SDK version API 22 and minimum SDK version being API 15 [16].

b) Java Development Kit 1.8 [15].

c) Xiaomi Redmi 1s [8] with a custom ROM (Resurrection Remix) [14] as the test device.

d) The application has been made on a desktop computer which has an i5(haswell), 16Gb of RAM, 128 GB SSD primary with 1TB HDD secondary. It was custom assembled and thus has no production name, with Ubuntu 14.04.2 as the default operating system.

D. *The Prototype*

The target domain list is stored in a universal location to allow easy scalability without requiring explicit application upgrades on all MAs, which would have been required to be done if the names had been hardcoded into the application.

1) *The Target List.*

The prototype requires the target domain names to be available and to allow easy scalability, the target domain names have been written and uploaded as a webpage. The webpage needs to be edited to allow the modification of the target list. The target domain names are written in tab-delimited format [17]. The list was chosen arbitrarily to the servers of the website which the DNS was redirecting to from Delhi, India.

2) *The Android Application.*

The application is packaged and installed on the test android smartphone. The application consists of a button and a blank listview, which is populated by the target domain names which are fetched from the website upon application starting. The list is stored locally as long as the application is running in the foreground. The application has been kept simple to ensure ease of use. The application screen-shot is shown in Figure 2.

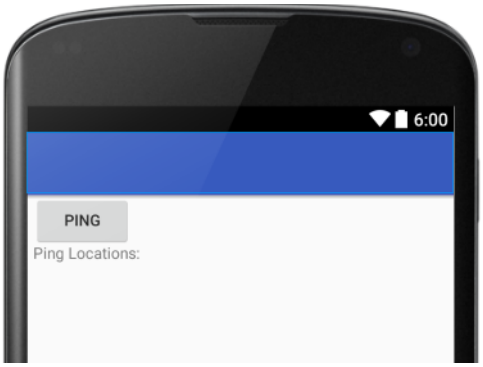

Figure 2: The Default Screen

After successful application installation, the domain names are fetched from the webpage and parsed to a listview format and presented to the user. The user can use the present "PING" button to initiate the measurement process. However the process requires root permission and the the user is requested to allow root request as shown in Figure 3. The user must accept this otherwise the application will not function.

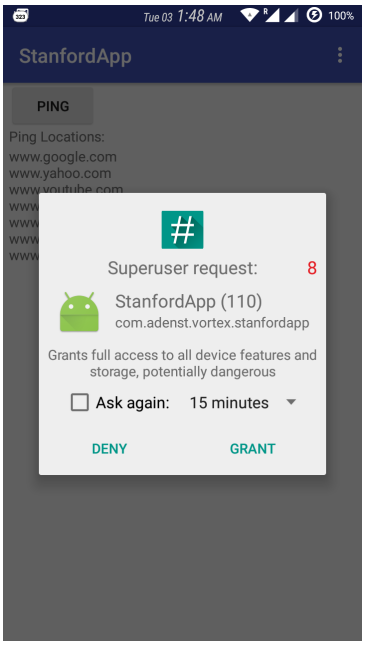

Figure 3: Root Permission Request

IV. OUTPUT

Upon application start, the application shall automatically scan the hardcoded url of the webpage for the list of target hostnames, and then it will fetch the content of the page, parse it and if successful will display to the user a message along with the list of target domain names as show in figure 4.

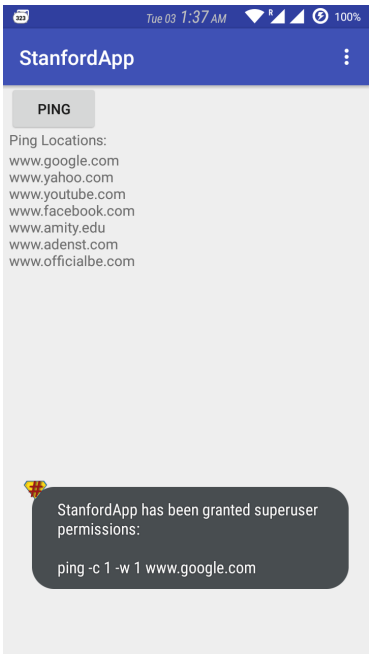

Figure 4: Successful Fetch of Targets

Upon user request through a button press the process starts and the target hosts are pinged individually and the individual requests are recorded in the Android Logcat. After successful/unsuccessful pings have taken place the data is stored in the application which can be used for further analysis if required. The log follows the format of date, time, application process ID and name, in-application process name, command executed, domain name, ip address, bytes of data sent as payload, as shown in figure 5.

11-03 01:37:15.320 12372-12372/com.adenst.vortex.stanfordapp E/MainActivity: PING www.google.com (173.194.36.115) 56(84) bytes of data.

Figure 5: Logcat information

In the application, data can be stored in many ways. The paper discusses three approaches that have been used.

A. Average RTT

This is the simplest of all three and was used to test the initial working of the application, as shown in figure 6. It gives limited data and not much scope for analysis.

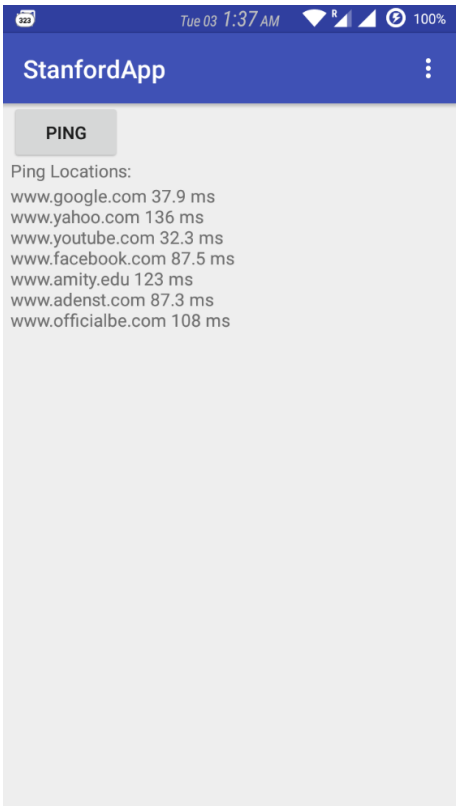

Figure 6: Average RTT data

B. Full RTT specification

This gives the minimum, average, maximum and the calculation of the mean deviation in a list format, and can be used for useful analysis purposes as shown in figure 7. The average RTT is shown beside the domain name.

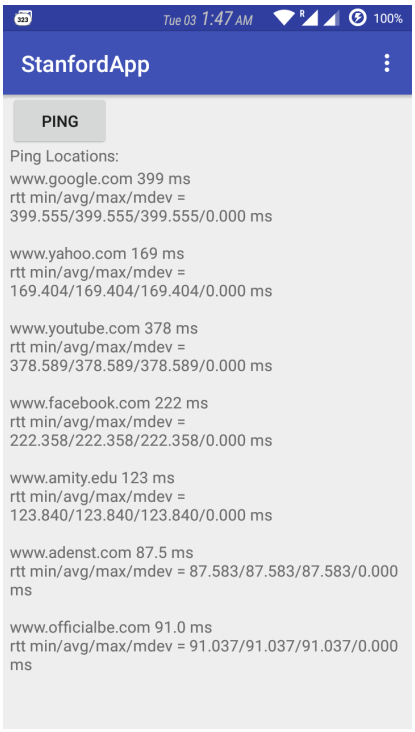

Figure 7: Full RTT Specification

C. Packet and RTT Specification

It allows for the receiving and storage of both packet information and full RTT specifics. The data is the most detailed and gives details about packet transmission delays, packet loss and also RTT information as section 4.2 does, as shown in figure 8.

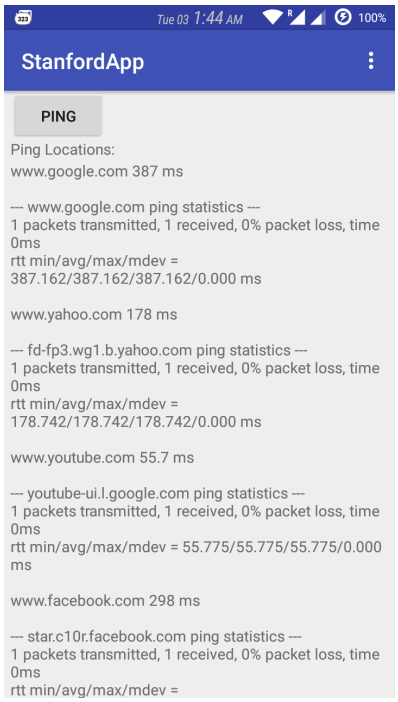

Figure 8: Full Packet and RTT Specific

V. Conclusion and Future Work

The application has been designed with an aim to emulate the activity of PingER software and hopes to be more space efficient and power efficient. Eventually this hopes to replace all desktop MAs and act as the most power efficient solution to the entire PingER process.

The design can be expanded to handle unpredictable situations, and also the data from many MAs may be uploaded to common servers to allow a central backup repository of the data. Also the performance of the application needs to be tested in different conditions.

References

[1] Hilbert, M. (2014). Technological information inequality as an incessantly moving target: The redistribution of information and communication capacities between 1986 and 2010. Journal of the Association for Information Science and Technology, 65(4), 821-835.

[2] Norris, P. (2001). Digital divide: Civic engagement, information poverty, and the Internet worldwide. Cambridge University Press.

[3] Thorvaldsen, Ø. E. (2006). Geographical location of internet hosts using a multi-agent system.

[4] Barry, B., Chukwuma, V., Petitdidier, N., Cottrell, L., & Barton, C. (2008, May). Digital divide in sub-Saharan African universities: Recommendations and monitoring. In IST-Africa Conference, Windhoek, Namibia (pp. 7-9)..

[5] Cottrell, R. L., Logg, C., Chhaparia, M., Gngonev, M., Haro, F., Nazir, F., & Sandford, M. (2006, April). Evaluation of techniques to detect significant network performance problems using End-to-End active network measurements. In Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP (pp. 85-94). IEEE.

[6] Cottrell, R. L., Logg, C., & Mei, I. H. (2003, April). Experiences and results from a new high performance network and application monitoring toolkit. In Passive and Active Measurement Workshop..

[7] Matthews W. and R. L. Cottrell, "The PingER Project: Active Internet Performance Monitoring for the HENP Community", IEEE Communications Magazine, May 2000.

[8] http://www.gsmarena.com/xiaomi_redmi_1s-6373.php

[9] Mukherjee, S., Prakash, J., & Kumar, D. (2015). Android Application Development & Its Security.

[10] Brady, P. (2008, May). Android anatomy and physiology. In Google IO developer conference.

[11] Cheng, B., & Buzbee, B. (2010, May). A jit compiler for android's dalvik vm. In Google I/O developer conference (Vol. 201, No. 0).

[12] https://www.griffith.edu.au/sustainability/sustainable-campuses/sustainable-initiatives/energy/average-computer-energy-usage

[13] http://www.forbes.com/sites/tristanlouis/2013/09/14/the-real-cost-of-a-smartphone/

[14]http://forum.xda-developers.com/redmi-1s/development/rom-resurrection-remix-lollipop-v5-3-9-t3069027

[15]http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

[16] http://developer.android.com/sdk/index.html

[17] http://www.adenst.com/aditya/

[18] Fall, K. R., & Stevens, W. R. (2011). TCP/IP illustrated, volume 1: The protocols. addison-Wesley.