Profiling Resource Utilization for Instant Messaging Applications

Harshit Shrivastava Rohit Gattani Sujith Shivaprakash

School of Informatics and Computing
Indiana University Bloomington, United States
hshrivas@indiana.edu, rgattani@iu.edu, sujishiv@iu.edu

ABSTRACT

Instant messaging (IM) applications have become the epicenter of our lives in the current era. While we use them day in and day out, we rarely pay any attention to check how much resources they might be using. In this study, we profiled instant messaging applications based on their energy consumption and network resource utilization on cellular network. We focused on two most popular instant messaging applications of the current era, WhatsApp and Facebook Messenger. This study gives us an overview of resource usage patterns of instant messaging applications, underlying problems, and available optimization opportunities. In our experiments, we have discovered network and radio resource usage abuse by so called 'tiny' features provided by instant messaging applications. We also analyze the differences in the resources utilization by WhatsApp and Facebook Messenger. We discuss some of the key limitations encountered during this study and recommend several optimization techniques.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design - Wireless Communications; C.4 [**Performance of Systems**]: Measurement Techniques.

General Terms

Design, Measurement, Performance.

Keywords

Instant messaging, Throughput, Network utilization, Battery consumption, RRC state machine, Facebook Messenger, WhatsApp.

1. INTRODUCTION

Instant messaging applications have already overtaken Short Message Service (SMS) and Multimedia Message Service (MMS) provided by cellular network operators. With recent advancements in smartphone technology and reduced mobile data costs, instant messaging applications have become widely popular across all age groups, garnering billions of daily active users. WhatsApp and Facebook Messenger are two such leading instant multimedia messaging service. Both applications have a user base of more than a billion. WhatsApp and Facebook Messenger are cross-platform mobile applications which allows users worldwide to instantly exchange text messages and multimedia contents such as photos, audio and videos. Activity on Facebook Messenger and WhatsApp combined is now three times the global volume of SMS messages, at 60 billion messages a day compared to 20 billion. [18]

With the recent surge in the usage of instant messaging applications, their wide popularity, and billions of active users, we set out to investigate the resource utilization by instant messaging applications. We explored and investigated two most popular instant messaging applications viz. WhatsApp and Facebook Messenger. We have profiled these IM applications based on their network and radio resource usage, battery consumption, and CPU utilization for various real-world user interaction patterns.

Some of our key observations are as follows: The traffic pattern of always-on-always-connected instant messaging applications is quite different from traditional internet applications. Instant messaging applications support real-time communication services, and are often constantly running in background to receive status updates, notifications, and messages from other users. Thus IM applications continuously exchange short signaling (keep-alive, ping, etc.) and data (text, multimedia, etc.) traffic with the network. Such frequent short messages incur a large amount of signaling traffic which further leads to rapid drainage of mobile device's battery and signaling storm at network end.

Although 'tiny' but features such as typing notification and online presence awareness updates lead to wastage of network and radio resources, and battery power. WhatsApp on an average takes ~70 bytes of data to notify if a person is online or offline. We observed that, WhatsApp consumed around 6KB of data to indicate if the user is typing a message or not over a period of ~80 seconds. Both WhatsApp and Facebook Messenger have highly efficient transcoding mechanism in place to compress and transfer multimedia content. Our analysis suggests that Facebook Messenger consumes approximately 1.4 times more battery power than WhatsApp while transferring same multimedia content.

2. RELATED WORK

Instant messaging applications have become an integral part of our daily lives, connecting hundreds of millions of users. Although much has been said and studied about the social aspects of using these IM applications [14] or how secure these applications are [15], little is known about the resource utilization. This is particularly due to the black box nature of these IM applications.

Applications like WhatsApp and Facebook Messenger have an active user base of more than a billion users each [16] [17]. While much of the work has been done in recent past on profiling instant messaging applications on the desktop and mobile devices, not much emphasis has been given to WhatsApp and Facebook Messenger.

Schrittwieser et al. [15] have evaluated the security models and authentication mechanisms of instant messaging applications. In [3], the focus is on networking and traffic characteristics, and content delivery infrastructure of popular online social networks. This study sheds some light on the issues such as usage patterns, content location, hosting organizations, and addressing dynamics. P. Fiadino et al. [1][2] have performed first large-scale characterization of WhatsApp revealing some facts about the hosting network architecture, characteristics of the network traffic, and the performance of media transfers. In [6], Y. Liu and L. Gao investigate the current practice of video messaging services on smartphones and reveal a few challenges in serving content without adequate network support such as a large amount of network traffic and upload/download latency.

E. Vergara et al. [5] study the interdependency between energy consumption and instant messaging application's data patterns. After analyzing the cost of added functionalities provided by IM applications and studying the user interaction patterns, this study suggests mechanisms to reduce energy consumption while retaining the usability. L. S. Meng et al. [7] investigate the power consumption due to the presence information exchange of instant messaging applications by examining various mechanisms and their implications, and propose solutions with some user experience tradeoff. Y. W. Chung [8] proposes an improved energy saving scheme for instant messaging services with two different inactivity timer values and compares the performance of proposed scheme.

M. Gupta et al. [13] investigate the impact of always-on-always-connected (chatty) mobile internet applications on LTE device power, air interface signaling, and quality of service requirements such as the latency constraints. In [12], Y. Choi et al. explore the traffic composition of 3G network traffic, study the signaling traffic surge caused by chatty mobile applications, and suggest the development of network-aware smart applications and notification services.

Previous studies have focused on analyzing the impact of emerging mobile internet applications on cellular networks and user devices. However, not much research has been done on network resource utilization, battery power consumption, radio resource usage, and processor utilization by instant messaging applications at the fine-grained use case level. We profile the resource utilization of instant messaging applications at use case level. Although being 'tiny', typing and online presence awareness notifications are crucial from a user's perspective, we explore and investigate such services provided by instant messaging applications.

3. BACKGROUND

3.1 Ecosystem

In this section we focus on the characterization of the network traffic and the content delivery infrastructure of Facebook and WhatsApp.

3.1.1 WhatsApp

A prior study by Fiadino et al. reveals that WhatsApp servers are associated with the domain name whatsapp.net and different third-level domain names are used depending on the nature of the message viz. control, chat, and media. Control and text messages are handled by chat servers on ports 5222 (XMPP) or 443, associated with the domain names {e|c|d}x.whatsapp.net where x is the variable for load balancing. Multimedia contents are handled by HTTPS multimedia servers identified by the domain names mms|mmi|mmvXYZ.whatsapp.net wherein mms|mmi for audio and photos, and mmv for videos. Each HTTPS connection is dedicated to a single content transfer and closed on transfer completion. Connections to chat servers are characterized by low data rate and long duration, whereas multimedia connections are short and transfer heavy flows.

WhatsApp is a fully centralized service, hosted by a single service provider with its cloud servers exclusively located in the US. All the messages are routed through the core network independent of the geographic location of the users. WhatsApp network traffic flows thus suffer from additional latency for users outside the US.

More than 380 server IPs owned by the cloud service provider SoftLayer serve the WhatsApp network traffic flows. WhatsApp keeps at least 200 servers active even in the lowest load hours. All the chat servers are constantly active to keep the state of active devices to quickly push messages. [1][2][3]

3.1.2 Facebook

Facebook uses a highly dynamic and distributed content delivery mechanisms. The content is hosted in multiple geographic locations on over 6500 server IPs and is provisioned through highly dynamic addressing mechanisms. Akamai serves almost 50% of the Facebook network traffic flows, using more than 2200 server IP's. Fiadino et al. reveal in their study that 65% of the total traffic volume is hosted by Akamai, followed by Facebook itself with about 19%. Akamai serves the static contents whereas, the Facebook almost exclusively serves the dynamic contents (e.g. chats, tags, session information, etc.). With the geographically distributed presence of servers, pushing contents as close as possible to end-users, latency due to propagation is highly reduced. [3][4]

3.2 RRC State Machine

In this section, we briefly explain the operation of RRC for further discussion.

When the device is first powered on, the user equipment is in de-registered state and indicates its existence to the network by performing registration. Long Term Evolution (LTE) power management model requires user equipment (UE) to be in either of two different operating modes: RRC_Connected and RRC_Idle. In the RRC_Connected mode, UE can listen to the network and transmit or receive data, and thus consumes higher power. After a period of inactivity (i.e. after inactivity timer expires), the radio link is released and UE transitions into RRC_Idle mode consuming much less power (i.e. in the range of few milliwatts). RRC inactivity timer is configured by network operators and typically is in the range of a few seconds to a few tens of seconds. Prior to data transmission, UE must reconnect to the network and go back to the RRC_Connected state in order to send or receive data.

Actually, the user equipment can be in one of four states: Idle, PCH, Cell_FACH, and Cell_DCH. In the Idle state, the UE does not maintain a connection with the network and is in low power state. In PCH state, UE is connected to the network but the UE cannot send any data and only checks for paging information. In Cell_FACH state the UE is in RRC_Connected state and can send or receive data over common or shared radio channels. This is an ideal state to send or receive short data packets. In Cell_DCH state the UE is connected to the network via a dedicated or high speed shared downlink radio channel. This state is an ideal state for the transfer of large data packets.

When the UE is in the RRC_Connected state, apart from time spent in data transfer, radio spends additional time waiting for the inactivity timer to expire before it goes back to the Idle state, and is often referred as tail time. Tail time results into significant wastage of energy and radio resources. Promotion from the Idle state to Cell_DCH/Cell_FACH state requires a lot of signaling between the UE and radio network controller (RNC) for resource allocation. Such promotion delays incur a long latency. [9][12][13]

4. PROBLEM DEFINITION

Instant messaging applications like WhatsApp and Facebook Messenger which have been primarily developed for smartphones are in predominant use in the recent times. Users spend roughly a third of their time using these instant messaging applications. However, smartphones suffer from resource constraints such as limited battery life and availability of network bandwidth. In this study, we profile instant messaging applications based on their energy consumption and network resource utilization. This study gives us an overview of resource usage patterns of instant messaging applications, underlying problems - origin and impact, and available optimization opportunities.

5. METHODOLOGY

We conducted experiments using three different Android smartphones between March 14, 2017 and April 24, 2017 over the commercial cellular network. Our study focuses on the two most popular instant messaging applications: WhatsApp and Facebook Messenger. Table 1 shows the versions of these applications that we installed on our testing devices.

Table 1: Instant messaging applications used

	Google Nexus 5	OnePlus X	Google Nexus 5X
WhatsApp	2.17.107	2.17.117	2.17.107
Facebook Messenger	110.0.0.14.69	110.0.0.14.69	110.0.0.14.69

In this study, we focus on the resource utilization of instant messaging applications at use case level. Instant messaging applications provide services such as sending and receiving text and multimedia messages, typing and online presence awareness notifications, and audio and video calling. In our study, we examine a subset of services provided by instant messaging applications. Our test cases imitate real-world user interactions with instant messaging applications. We ask, 'How long it takes the user to upload or download a message', 'How many bytes are transferred over the uplink and downlink channels', 'What are the throughput and latency values', 'How much radio energy is consumed, and what is the impact on battery power consumption'. Our experimental methodology is straightforward. For each application studied, we collected traces by using the application as specified in our designed test cases. We then analyzed the trace using ARO and Wireshark.

We explored and investigated various tools to capture and analyze network traffic. Tools like 'Tpacket capture' and 'Packet capture' can be easily downloaded from Android Google Play Store. However, these applications require establishing the virtual private network on the mobile

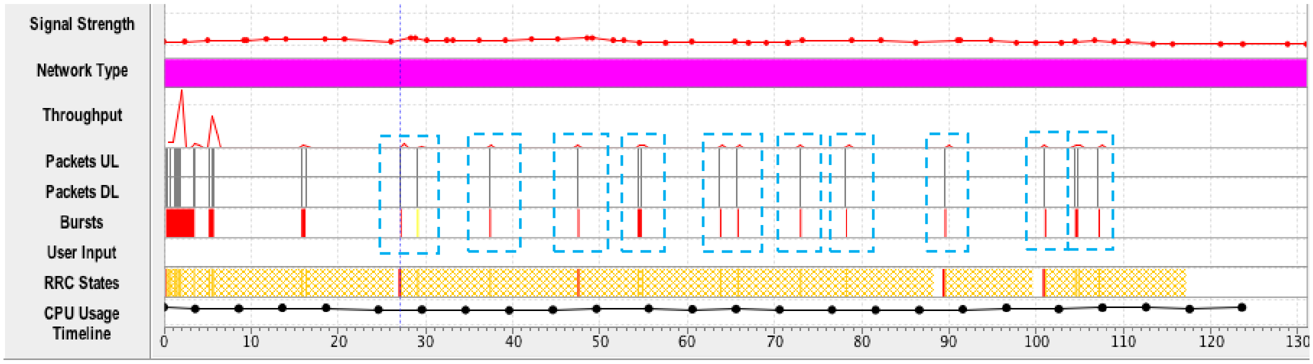

Figure 1: WhatsApp-Typing notification trace (each dotted box represents a typing notification burst)

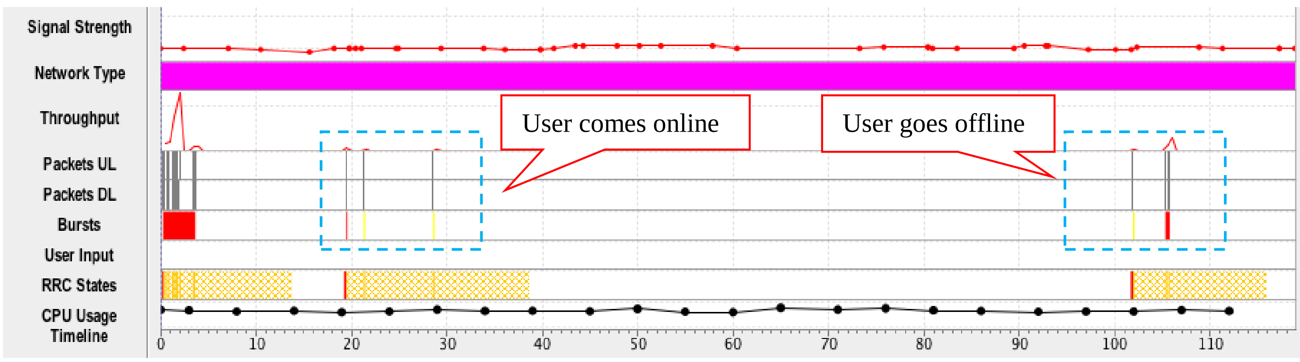

Figure 2: WhatsApp-Online presence awareness trace – receiver's perspective (WhatsApp)

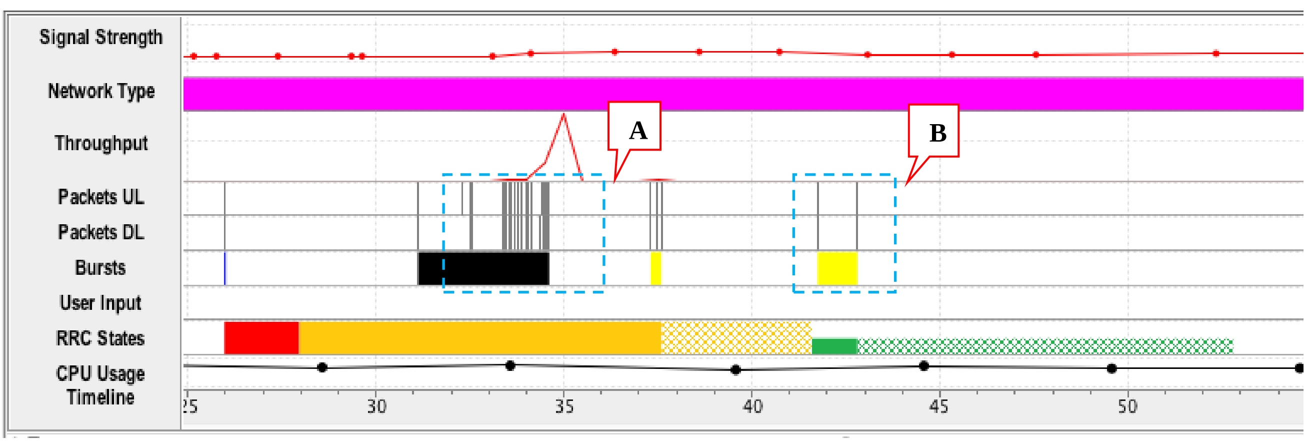

Figure 3: WhatsApp-Image transfer trace (A: TCP connection established and image data transferred, B: Read receipt received)

device to monitor all the incoming and outgoing network traffic. Since the credibility of the above applications' owners could not be verified, we decided to not to use these tools for the study. 'Tcpdump' on the other hand requires the mobile device to be rooted, which voids the device manufacturer warranty. Hence, using tcpdump to capture and analyze network traffic was out of scope. Another well-known tool 'Wireshark' is best in class to capture and analyze network traffic, but it is verbose and lacks a cross-layer approach, which is provided by AT&T's Video Optimizer (ARO). ARO exposes the cross layer interaction between the layers ranging from radio resource control to application layer. We thus capture and analyze the trace using ARO and use Wireshark for secondary analysis. We also used 'Trepn', an android application developed by

Qualcomm to efficiently measure the energy usage by IM applications.

6. FINDINGS

In this section, we briefly discuss some of the key findings of our study. We discuss how typing and online presence awareness notifications lead to wastage of network and radio resources, and battery power. We also discuss the resource utilization statistics of transferring an image using IM application.

6.1 Typing Notification

The typing notification is one of the most common features implemented by instant messaging applications. This feature allows the user to be notified when the other user(s) in the conversation is (are) typing, thus creating a notion of presence and interactivity. When the user (sender) starts typing in the text input area of the graphical user interface (GUI) of IM application, the other user (receiver) is immediately notified. Normally, this features allows the user to decide whether to remain active in the conversation or not.

In our test case for typing, we asked the user to open the conversation window of instant messaging application and start typing two or three meaningful sentences in text input area of GUI. We asked the user to not to send the typed text and later on erase the typed text. Figure 1 shows the corresponding trace collected using ARO.

When the user opens conversation window, application establishes a connection to the WhatsApp chat and control server (IP 108.168.176.236, on port 5222) owned by SoftLayer Technologies Inc. (registered for organization WhatsApp Inc.). As user starts typing, client application initiates network traffic, exchanging 85 bytes over 8 TCP packets with throughput of 27.2kbps. As user continues typing for next 80 seconds, client initiated network traffic transfers approximately 6KB worth of data as part of typing notification update. Preliminary analysis suggests that after every 8 seconds WhatsApp sends a typing notification. The typing notification feature keeps UE radio in RRC_Connected (active) state for almost the entire duration of test.

We find that even though user (sender) never sent even a single character to receiver in reality, 6KB of data was exchanged as part of typing notification updates over a span of 80 seconds, and approximately 100J of radio energy was spent in high bandwidth, high energy state looking for packets. This analysis clearly suggests that although 'tiny' but typing notification feature incurs high cost in terms of radio energy consumption and network resource utilization. A study done by E. Vergara et al. [5] suggests that typing functionality increases the energy consumption by 40-104%.

6.2 Online Presence Awareness

Another common feature implemented by the instant messaging application is online presence awareness. This feature provides the instantaneous online status of other users of the instant messaging application.

We designed a test case where we asked the user to open a conversation window of instant messaging application and stay online for a minute or so, and later on close the application and switch to the home screen. At the same time, we asked the second user to open a conversation window with the first user and monitor his online presence status. Figure 2 shows the trace collected from second user's (receiver's) perspective.

When the first user comes online, the second user witnesses server-initiated traffic exchanging 34 bytes of data over 2 TCP packets from server IP 169.45.214.231 on port 5222. When the first user goes offline after ~70 seconds, the second user again witnesses server-initiated traffic exchanging 38 bytes of data over 2 TCP packets from same server IP. The second user receives these two bursts of data as part of online presence awareness updates for the first user.

Although it could not be confirmed, but we believe that WhatsApp implements an event-triggered periodic presence update mechanism wherein user notifies its presence to the control server and control server then immediately sends an online presence awareness notification to the all subscribed users.

As observed in the case of typing notification updates, the user equipment is forced to stay in high bandwidth, high energy state looking for packets until inactivity timer expires (also known as long DRX tail). We conclude that while effectively negligible data transfer took place in these two data bursts, UE radio was forced to switch to RRC_Connected state and wait for inactivity timer to expire before it could return to the low power RRC_Idle state. Online presence awareness updates, thus, not only exacerbate the limited battery power situation but also burden the cellular network infrastructure causing a surge in signaling traffic. Same is true for previously discussed typing notifications.

6.3 Image Transfer

WhatsApp implements a store-and-forward message delivery mechanism. To send a message to the recipient(s), the message is first uploaded to the server. The server then sends a notification message to the receiver. Once the receiver is notified, it downloads the message from the server. [6] As part of this study, we asked the user to transfer an image using instant messaging application. Figure 3 shows the trace collected using ARO.

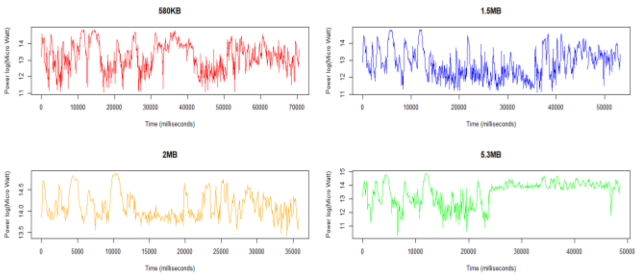

Figure 4: Power consumption of Facebook Messenger in log(microwatt) scale and time in milliseconds

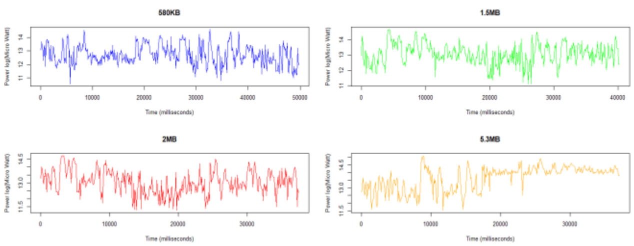

Figure 5: Power consumption of WhatsApp Messenger in log(microwatt) scale and time in milliseconds

Before sending an image, WhatsApp client application establishes a TCP connection with identified multimedia server (157.240.2.53 on port 443). Once the connection is established with the server, client application uploads the image by exchanging ~310KB of data over 477 TCP packets in roughly ~3.461s. Application spent 13 seconds compressing the image and exchanging metadata with the control server before actual image content is uploaded to the server. Once all the intended recipients have downloaded the image, the sender receives a read receipt notification through server-initiated traffic comprising of 300 bytes of data.

Our study reveals that WhatsApp has implemented an efficient transcoding mechanism for multimedia content. In our trace collection, we observed that original image of size 1.4MB was compressed to ~310KB. Effectively reducing the multimedia content benefits the user in terms of reduced mobile data costs, latency, and energy consumption. It also benefits cellular network operators by lowering the signaling traffic.

We have also analyzed the power consumption by Facebook Messenger and WhatsApp for image transfers. Figure 4 and Figure 5 depict plots of energy consumed by Facebook and WhatsApp respectively in log(microwatt) scale and time in milliseconds. We asked the user to transfer 4 different images of varying size (580KB, 1.5MB, 2MB, and 5.3MB) and recorded the measurements as an average of 4 iterations. We observe that as the image size increases, the power consumption also goes up. This observation can be attributed to the fact that increase in the size of image increases the number of bytes to be uploaded and time to upload, thereby consuming a more energy for a relatively longer period of time. Our analysis reveals that the average energy consumed by Facebook messenger is 1.4 times higher than WhatsApp.

7. LIMITATIONS

Although we tried to make sure that the work done as part of this study is comprehensive, there were certain aspects which we could not overcome. In this section, we enlist and elaborate on the limitations of this study.

First, all the data collection was done manually, and no automation tool was used for collecting the data. We could only collect a small number of traces since it took us a long time to collect each trace. Often we were forced to discard the collected traces due to synchronization issues, user behavior randomness, poor signal strength, etc. We collected a total 33 sets of traces during the course of our study. We believe that automation of experiments would have got us a large number of sets of traces. Also, current test cases imitate primitive user interactions while using instant messaging applications. A large scale automated measurement study is required to better understand the resource utilization of IM applications.

Second, we were unable to analyze the secured network traffic sent over HTTPS. Since we collected the traces by connecting the mobile devices with the laptop and capturing data by establishing VPN, there were obvious cases of HTTP Public Key Pinning (HPKP), commonly known as certificate pinning. HPKP is a security feature that allows the web server to safely associate a particular web client with a specific cryptographic public key. HPKP decreases the risk of man in the middle (MITM) attacks which primarily occur due to the forged certificates. We were unable to profile Facebook Messenger precisely due to HPKP security feature.

Third, it is difficult to get exact statistics on CPU utilization due to a variety of reasons. We noticed that there were different readings for CPU utilization for the same app across phones, models, and units. In the modern mobile devices, multi-core processors (which is, in fact, a combination of low power and high power cores) are used. The goal of the operating system on any mobile device is to optimize CPU utilization and battery power consumption. Hence CPU allocates low power cores first, followed by high power cores. It is highly impossible to get CPU utilization values which would be consistent across applications, manufacturers, units, processors, and chipsets. Hence, we might get different readings of CPU utilizations by an application, captured on a particular phone at two different times. Due to this highly variable nature, we were unable to clearly estimate the CPU utilization of WhatsApp and Facebook Messenger.

Despite the above mentioned limitations, we collected a rich set of traces that led us to few very interesting observations.

8. RECOMMENDATIONS

Based on our experiments, evaluations, and findings, we came up with few recommendations that we are recording in this section.

When developing energy efficient instant messaging applications there is a trade-off between application features and resource utilization. There is a need to thoroughly analyze the impact of adding these features to network and radio resource usage, battery power consumption, CPU utilization and quantify the perceived benefits to the user.

Our study shows that typing notification feature provided by instant messaging applications leads to tremendous energy consumption, a surge in signaling traffic, and wastage of network bytes. We recommend that instant messaging applications reduce the frequency of typing notifications. E. Vergara et al. [5] conclude that the message patterns of the conversation greatly influence the energy consumption. In dense conversations, the impact of typing notifications is not as high as in the sparse conversations. Application developers should also provide an option to dynamically enable or disable typing notifications.

Our study shows that sending or receiving text messages trigger network transmission and shows resource utilization behavior similar to typing notification. We recommend a policy of bundling which can be dynamically activated in case of sparse conversations or based on the user activity. If the receiver is online and active in the chat, the messages should be sent immediately to the receiver. However, if the receiver is away or not active in the conversation, then instead of immediately triggering network transmission, exchanging messages can be deferred and bundled with the other messages. Transferring messages in the same packet reduce the amount of protocol overhead evident in short and frequent network traffic. Bundling also allows UE to stay in RRC_Idle state for a longer period and provides a better opportunity for compression.

While transcoding and encoding lower the resource utilization by instant messaging applications, we recommend that users be given greater flexibility when it comes to recording and transmission quality since, after all, the user experience is paramount.

[3] suggests and we recommend that ISP and Cellular network providers can better optimize the network resources through time-based traffic engineering mechanisms, dynamically adjusting network resources based on load predictions for emerging mobile internet applications (particularly instant messaging applications).

9. CONCLUSION

With the recent surge in the usage of instant messaging applications, their wide popularity, and billions of active users, we set out to investigate the resource utilization by instant messaging applications. We explored and investigated two most popular instant messaging applications viz. WhatsApp and Facebook Messenger. We profiled these IM applications based on their network and radio resource usage, battery consumption, and CPU utilization for various real-world user interaction patterns. During the analysis of collected traces, we discovered few very interesting facts. In this paper, we have summarized our experiments, findings, recommendations, and limitations of our study. Our study shows that characterizing instant messaging applications is paramount for cellular network operators and application developers to better optimize resource usage at both the ends - user and network.

10. ACKNOWLEDGMENTS

We sincerely thank Prof. Feng Qian (Indiana University Bloomington) for guiding us throughout the course of this work. We also appreciate Prof. Qian for his thorough feedback and suggestions, which were crucial to finding solutions to our problems.

11. REFERENCES

[1] Pierdomenico Fiadino, Mirko Schiavone, Pedro Casas. Vivisecting WhatsApp through Large-Scale Measurements in Mobile Networks. In SIGCOMM, 2014.

[2] Pierdomenico Fiadino, Mirko Schiavone, Pedro Casas. Vivisecting WhatsApp in Cellular Networks: Servers, Flows, and Quality of Experience. To appear in TMA, 2015.

[3] Pierdomenico Fiadino, Pedro Casas, Mirko Schiavone, Alessandro D'Alconzo. Online Social Networks Anatomy: on the Analysis of Facebook and WhatsApp in Cellular Networks.

[4] Erik Nygren, Ramesh K. Sitaraman, Jennifer Sun. The Akamai Network: A Platform for High-Performance Internet Applications. In ACM SIGOPS 44(3), 2010.

[5] Ekhiotz Jon Vergara, Simon Andersson, Simin Nadjm-Tehrani. When Mice Consume Like Elephants: Instant Messaging Applications. In e-Energy, 2014.

[6] Yao Liu, Lei Gao. An Empirical Study of Video Messaging Services on Smartphones. In NOSSDAV 2014.

[7] Ling-San Meng, Da-Shan Shiu, Ping-Cheng Yeh, Kuan-Chi Chen, Hung-Yi Lo. Low power consumption solutions for mobile instant messaging. IEEE Transactions on Mobile Computing, 11(6):896–904, June 2012.

[8] Yun Won Chung. An Improved Energy Saving Scheme for Instant Messaging Services. IEEE Wireless Advanced.

[9] F. Qian, Z. Wang, A. Gerber, Z. Mao, S. Sen, and O. Spatscheck. Profiling resource usage for mobile applications: A cross-layer approach. In Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services, MobiSys ’11, pages 321–334. ACM, 2011.

[10] F. Qian, J. Huang, J. Erman, Z. M. Mao, S. Sen, and O. Spatscheck. How to reduce smartphone traffic volume by 30%? In Proceedings of the 14th International Conference on Passive and Active Measurement, IMC ’13, pages 42–52. Springer-Verlag, 2013.

[11] F. Qian, Z. Wang, Y. Gao, J. Huang, A. Gerber, Z. Mao, S. Sen, and O. Spatscheck. Periodic transfers in mobile applications: network-wide origin, impact, and optimization. In Proceedings of the 21st International Conference on World Wide Web, WWW ’12, pages 51–60. ACM, 2012.

[12] Yongmin Choi, Cha-hyun Yoon, Young-sik Kim, Seo Weon Heo, John A. Silvester. The impact of application signaling traffic on public land mobile networks. In IEEE Communications Magazine, January 2014.

[13] Maruti Gupta, Satish C. Jha,Ali T. Koc, Rath Vannithamby. Energy impact of emerging mobile internet applications on LTE networks: issues and solutions. In IEEE Communications Magazine, February 2013.

[14] R. E. Grinter, L. Palen, M. Eldridge. Chatting with teenagers: Considering the place of chat technologies in teen life. ACM Transactions on Computer-Human Interaction, Dec. 2006.

[15] S. Schrittwieser, P. Fr• uhwirt, P. Kieseberg, M. Leithner, M. Mulazzani, M. Huber, and E. R. Weippl. Guess who's texting you? evaluating the security of smartphone messaging applications. In 19th Annual Network and Distributed System Security Symposium (NDSS), ISOC. 2012.

[16] https://www.statista.com/statistics/260819/number-of-monthly-active-whatsapp-users Statista. Retrieved 2017-04-23.

[17] https://www.statista.com/statistics/417295/facebook-messenger-monthly-active-users Statista. Retrieved 2017-04-23.

[18] https://www.theverge.com/2016/4/12/11415198/facebook-messenger-whatsapp-number-messages-vs-sms-f8-2016. Retrieved 2017-05-01.