



รายงานวิจัยฉบับสมบูรณ์

โครงการระบบการรับส่งข้อความและรูปภาพที่เป็นความลับด้วยโทรศัพท์มือถือสำหรับ
หน่วยงานที่ต้องการความมั่นคงของข้อมูล

ณัฐนนท์ ลีลาตระกูล
ภาณุจ รัตนวรพันธ์
เอกภพ บุญเพ็ญ

โครงการวิจัยประเภทงบประมาณเงินรายได้
จากเงินอุดหนุนรัฐบาล (งบประมาณแผ่นดิน)
ประจำปีงบประมาณ พ.ศ. ๒๕๖๐

รายงานวิจัยฉบับสมบูรณ์

โครงการการรับส่งข้อความและรูปภาพที่เป็นความลับด้วยโทรศัพท์มือถือสำหรับ
หน่วยงานที่ต้องการความมั่นคงของข้อมูล

ณัฐนนท์ สีลาตระกูล
ภาณุจ รัตนวรพันธุ์
เอกภาพ บุญเพ็ง

กิตติกรรมประกาศ

งานวิจัยนี้ได้รับทุนสนับสนุนการวิจัยจากงบประมาณเงินรายได้ จากเงินอุดหนุนรัฐบาล (งบประมาณแผ่นดิน) ประจำปีงบประมาณ พ.ศ. 2560 มหาวิทยาลัยบูรพา ผ่านสำนักงานคณะกรรมการการวิจัยแห่งชาติ เลขที่สัญญา 163/2560

Acknowledgment

This work was financially supported by the Research Grant of Burapha University through National Research Council of Thailand (Grant no. 163/2560).

บทคัดย่อ

ปัจจุบันมีโปรแกรมส่งข้อความตอบโต้ในเวลาจริง (messaging application หรือ chat application) บนโทรศัพท์มือถือให้ผู้ใช้ได้เลือกใช้งานอยู่เป็นจำนวนมาก โปรแกรมเหล่านี้ส่วนใหญ่จะมีเซอร์เวอร์กลางเป็นตัวเชื่อมระหว่างผู้ใช้งานที่ส่งข้อมูลกันไปมา ผู้ใช้งานให้ความเชื่อถือ (trust) กับระบบเซอร์เวอร์กลางว่าจะเป็นผู้รักษาความปลอดภัยและความเป็นส่วนตัว (security and privacy) ของข้อมูลผู้ใช้ ทั้งที่จัดเก็บหรือที่อยู่ในระหว่างการส่งต่อจากผู้ใช้งานไปสู่เซอร์เวอร์กลางและในทางตรงข้าม ได้เป็นอย่างดี อย่างไรก็ตาม messaging app ในลักษณะนี้ไม่เหมาะกับหน่วยงานที่ต้องการความมั่นคงของข้อมูล ซึ่งไม่สามารถจะ trust บุคคลที่สาม (third party) ที่เป็นผู้ให้บริการ app เหล่านี้ได้ ดังนั้น messaging app ที่จะนำมาใช้ในสถานการณ์เช่นนี้ จะต้องไม่มี trusted third party เข้ามาเกี่ยวข้อง กล่าวคือต้องทำการเข้ารหัสแบบต้นถึงปลาย (end-to-end encryption)

คณะผู้วิจัยเลือกที่จะสร้างและทดสอบ end-to-end encryption messaging app โดยใช้ฐานโค้ดจากโครงการ Signal และเลือกแพลตฟอร์มในการพัฒนาบนมือถือแอนดรอยด์ (<https://github.com/signalapp/Signal-Android>) ทั้งนี้เพราะระบบมือถือแอนดรอยด์ครอบครองตลาดมือถือมากกว่า 70% (จากรายงานของ Gartner ที่เป็นบริษัทที่ปรึกษาเทคโนโลยีสารสนเทศรายใหญ่) และ Signal เป็น end-to-end encryption messaging app ที่เป็นเผยแพร่โปรแกรมซอร์ส (open-source) ซึ่งทำให้เราสามารถทดสอบ แต่งเติม และยืนยัน คุณสมบัติ end-to-end ตามที่ Signal ได้กล่าวอ้างไว้ได้จากการทดลองติดตั้ง ทดสอบ Signal เซอร์เวอร์ที่เราได้ทำการแต่งเติมบนเซอร์เวอร์ของ DigitalOcean และใช้ Minio เป็นตัวเก็บข้อมูลชั่วคราวบนคลาวด์ พบว่า Signal มีสมรรถนะดีพอใช้และสามารถเพิ่มขยายเพื่อรองรับผู้ใช้งานจำนวนมากขึ้นได้โดยง่าย (scalable)

Abstract

In today's world of mobile phone's ubiquity, there are many messaging applications (or chat applications) from which users can choose to communicate instantly online. Most of these applications offer some level of security and privacy, such as, in-transit and at-rest encryption of data, if the users are willing to trust centralized servers that host them. However, in communicating highly confidential data that have national security at stake, there cannot be trusted third parties like those central servers. In this case, end-to-end encryption is required.

In this project, we build and test an end-to-end encryption messaging app using codebase from Signal (<https://github.com/signalapp/Signal-Android>). Android is our platform of choice because of its dominating presence. According to a report from Gartner, a reputable IT consulting firm, Android has over 70% market share. Signal is an open-source messaging app that offers the end-to-end encryption feature. We need this open-source-ness because we want to modify and verify the source code to make sure it behaves according to specification. We have deployed our modified Signal app on DigitalOcean servers using cloud storage Minio for data buffering and found that Signal offers acceptable performance. In addition, it is also scalable. We can increase the number of servers to handle more users with little modification to the code for single server.

สารบัญ

บทที่ 1	
บทนำ.....	1
1.1 เนื้อหาของเรื่องที่เคยมีผู้ทำการวิจัยมาก่อน.....	1
1.2 ความสำคัญและที่มาของปัญหา	1
1.3 วัตถุประสงค์.....	1
1.4 ขอบเขตการวิจัย	1
1.5 แนวความคิดที่นำมาใช้ในการวิจัย.....	2
1.6 ประโยชน์ที่คาดว่าจะได้รับ	2
บทที่ 2	
เนื้อเรื่อง.....	3
2.1 รายละเอียดเกี่ยวกับวิธีดำเนินการวิจัย	3
2.2 ผลการวิจัย.....	3
บทที่ 3	
อภิปรายผลการวิจัย.....	88
บทที่ 4	
สรุปและข้อเสนอแนะ	90
บทที่ 5	
ผลผลิต	91
บรรณานุกรม.....	92

บทที่ 1

บทนำ

1.1 เนื้อหาของเรื่องที่เคยมีผู้ทำการวิจัยมาก่อน

ขอเป็นส่วนนี้ไปกล่าวไว้ในภาคผนวก

1.2 ความสำคัญและที่มาของปัญหา

ปัจจุบันหน่วยงานต่าง ๆ ในประเทศไทย ที่เกี่ยวข้องกับข้อมูลที่ต้องการความมั่นคง ปลอดภัย สูงเช่น กองทัพบกยังไม่มี messaging app ที่ใช้สำหรับการสื่อสารบนสมาร์ตโฟนที่มีคุณสมบัติการทำ end-to-end encryption ของหน่วยงานเอง GChat ที่มีใช้อยู่และได้รับการส่งเสริมจากรัฐบาลใช้ trusted third party ที่มีศูนย์กลางที่สามารถถอดรหัสข้อมูลได้ ดังนั้นจึงมีความเสี่ยงเป็นอย่างมากถ้าจะส่งผ่านข้อมูลที่มีเกี่ยวข้องกับ ความมั่นคงของประเทศผ่านช่องทางนี้

ในโครงการวิจัยนี้เราจึงได้สร้างและทดสอบ messaging app ที่มีคุณสมบัติ end-to-end encryption โดยใช้ codebase จากโครงการ Signal สำหรับ Android โดยมีจุดประสงค์หลักเพื่อให้หน่วยงานที่เกี่ยวข้องกับ ข้อมูลที่ต้องการความลับสูงสุดนำไปประเมินและใช้งานกันในวงกว้างต่อไป

ผู้วิจัยเลือก Signal เพราะมีความเป็น open-source ทำให้เราสามารถปรับแต่งและพิสูจน์ความถูกต้อง ของโปรแกรมที่จะใช้งานจริงได้ และเราเลือก Android เพราะเป็นแพลตฟอร์มที่โดดเด่นและใช้งานกันมากในวง กว้าง อย่างไรก็ตามแนวทางการวิจัยและวิธีปฏิบัติที่เราใช้ในโครงการนี้ มีความยืดหยุ่นที่ทำให้เราสามารถนำไปใช้ กับ codebase และแพลตฟอร์มอื่นๆได้

1.3 วัตถุประสงค์

1. เพื่อสร้างและทดสอบ messaging application ที่มีความสามารถในการทำ end-to-end encryption
2. เพื่อส่งเสริมให้มีการใช้งาน messaging application ที่สร้างขึ้นในองค์กรของรัฐที่เกี่ยวข้องกับข้อมูลที่มี ผลต่อความมั่นคงของประเทศ อย่างแพร่หลาย

1.4 ขอบเขตการวิจัย

โครงการวิจัยนี้เป็นโครงการที่เกี่ยวข้องกับการพัฒนาและทดสอบซอฟต์แวร์เป็นหลัก โดยมีขอบเขตของ การพัฒนาและการทดสอบดังนี้

1. การพัฒนาซอฟต์แวร์ไม่ได้เริ่มต้นจากศูนย์แต่ใช้ codebase จากโครงการ Signal ที่เป็นโครงการ open-source ทำให้เราสามารถปรับแต่ง แก้ไข และทดสอบ ความถูกต้องในระดับ source code ได้
2. เป้าหมายแพลตฟอร์มสำหรับซอฟต์แวร์ที่สร้างขึ้นคือ Android สมาร์ทโฟนเพราะเป็นแพลตฟอร์มที่มีความแพร่หลายและได้รับการยอมรับมากกว่าแพลตฟอร์มอื่น ๆ มาก (มีส่วนแบ่งการตลาดมากกว่า 70%)
3. สำหรับโครงการในเฟสแรก เราเน้นการติดตั้งและทดสอบเริ่มต้นเพียงหนึ่งเซิร์ฟเวอร์ โดยเราคาดหวังว่าจะสามารถขยายเพิ่มเป็นหลายเซิร์ฟเวอร์ได้โดยไม่ยากและสามารถใช้ codebase เดียวกันมาปรับแต่งเพียงเล็กน้อยเท่านั้น

1.5 แนวความคิดที่นำมาใช้ในการวิจัย

งานวิจัยนี้เป็นการพัฒนาและทดสอบซอฟต์แวร์ที่มีวงจรการพัฒนา (software development life cycle หรือ SDL) เป็นไปตามกระบวนการต่อไปนี้ การสำรวจ วางแผน และออกแบบ (planning and design) การเขียนโปรแกรม (coding) และการทดสอบ (Testing)

กระบวนการเหล่านี้สามารถทำวนซ้ำได้ (iterative process) กล่าวคือหลังจากการทดสอบแล้ว อาจกลับไปแก้ไขในส่วนของการออกแบบหรือการเขียนโปรแกรมอีกครั้ง ถ้าผลการทดสอบยังไม่เป็นที่พอใจ โดยในทางปฏิบัติการกระทำวนซ้ำเป็นเรื่องปกติ แทบจะไม่มีโครงการใดที่เกี่ยวข้องกับซอฟต์แวร์ขนาดใหญ่แล้วจะผ่านกระบวนการทั้งหมดนี้เพียงหนึ่งรอบกระบวนการ

1.6 ประโยชน์ที่คาดว่าจะได้รับ

1. หน่วยงานของรัฐที่เกี่ยวข้องกับข้อมูลที่ต้องการรักษาความลับในระดับสูงได้มี messaging app ที่มีคุณสมบัติ end-to-end encryption ไว้ใช้งาน ลดความเสี่ยงในการรั่วไหลของข้อมูลที่มีเดิมพันสูงได้มาก
2. คณะผู้วิจัยได้เรียนรู้เทคนิคล่าสุดในการพัฒนา messaging app ที่มีความปลอดภัยสูงสุด สามารถนำไปประยุกต์กับงานที่มีความเกี่ยวข้อง ใกล้เคียงกันได้
3. นิสิต นักศึกษา ผู้ร่วมโครงการ ได้เรียนรู้แนวทางการทำวิจัย รวมไปถึงทฤษฎีและเทคโนโลยีล่าสุดในการสร้าง secure messaging app

บทที่ 2

เนื้อเรื่อง

2.1 รายละเอียดเกี่ยวกับวิธีดำเนินการวิจัย

เราจะดำเนินการเป็น 2 ขั้นตอนหลักดังต่อไปนี้

1. ประเมิน messaging app ที่มีใช้งานอยู่ในปัจจุบันอย่างละเอียดก่อนที่จะเลือก Signal มาใช้ในการวิจัยในขั้นต่อไป
2. ทำการแต่งเติม ติดตั้ง ทดสอบ Signal บน DigitalOcean โดยใช้ Minio cloud storage และ รายงานผล

2.2 ผลการวิจัย

ต่อไปนี้จะรายงานการศึกษาการใช้งาน messaging app ต่อไปนี้อย่างละเอียด LINE Skype WhatsApp Telegram Signal และ Snapchat

ผลการเปรียบเทียบ messaging application

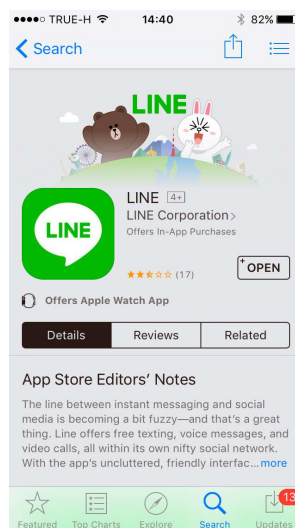
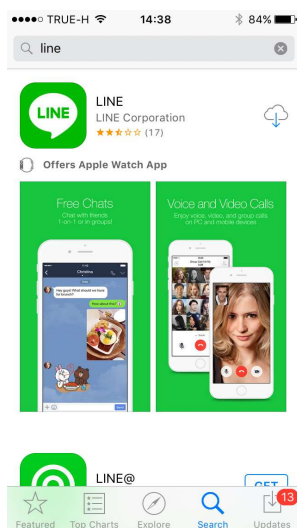
LINE



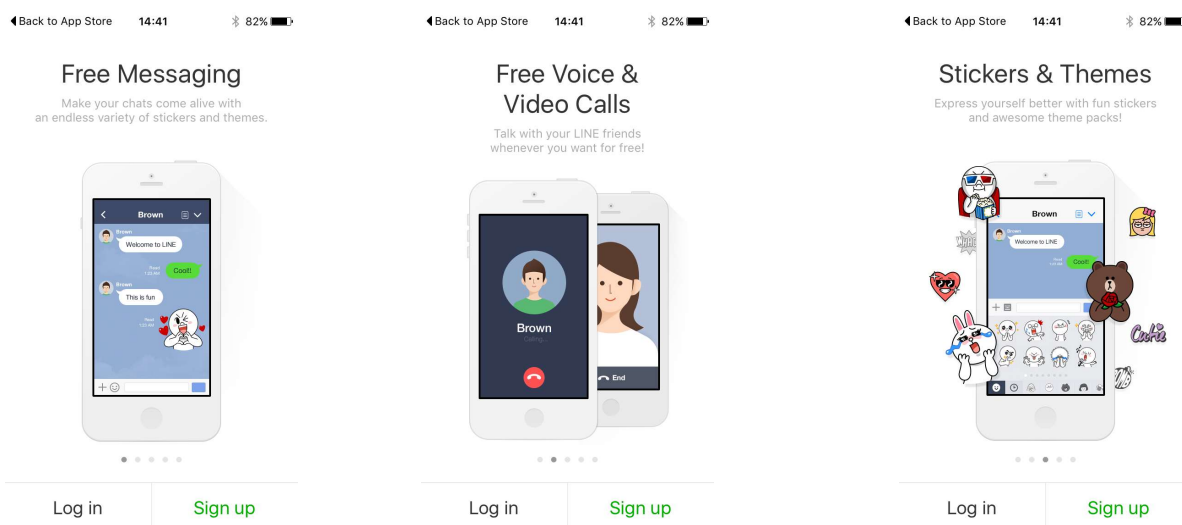
เป็นโปรแกรมเมสเซนเจอร์ที่สามารถใช้งานได้ทั้งบนโทรศัพท์มือถือที่มีระบบปฏิบัติการ ios , android , Windows Phone และสามารถใช้งานได้บนคอมพิวเตอร์ส่วนบุคคล และ Mac OS ด้วยความที่มีลูกเล่นมากมาย สามารถคุย(ทั้งในรูปแบบข้อความและเสียง) ส่งรูป ส่งวิดีโอ ส่งไอคอน ส่งสติ๊กเกอร์ ตั้งค่าคุยกันเป็นกลุ่ม ฯลฯ โดยความโดดเด่นของLINEไม่ได้มีเพียงความสามารถที่ทำให้หลากหลาย แต่ยังมี LINE Sticker และ LINE Friend(ตัวการ์ตูนที่ออกแบบโดยLINE) ซึ่งมีความน่ารัก ดึงดูดใจผู้ใช้งานอีกด้วย

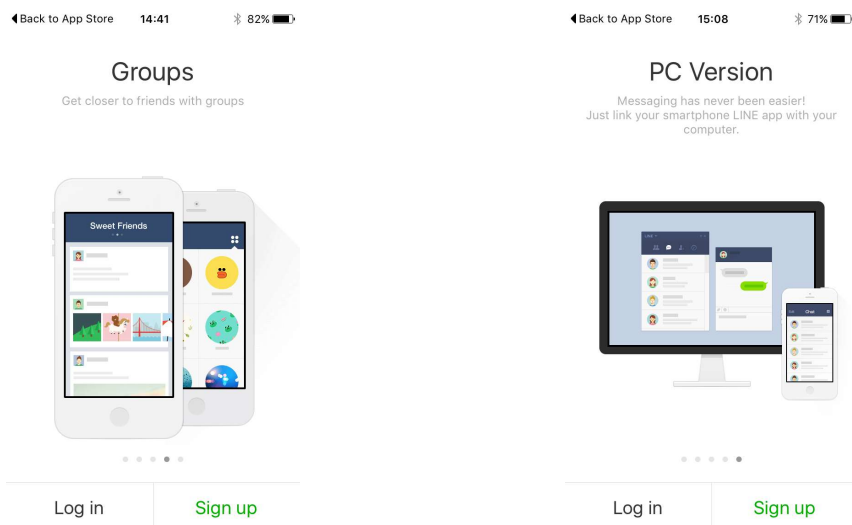
LINE บน iOS

เมื่อเราต้องการใช้งานLINEบนios เราสามารถเข้าไปdownloadได้จากApp Store และทำการติดตั้งบนเครื่องได้ทันที



หลังจากทำการติดตั้งLINEเรียบร้อยแล้ว เราlog inจากaccountที่มีอยู่ได้เลยหรือหากยังไม่มีaccountก็สามารถสมัครได้จากตรงนี้เลย ซึ่งในหน้าlogin หรือ sign upนี้ จะมีการแสดงคำบรรยายเกี่ยวกับความสามารถหลักของ application LINE โดยสรุปให้เราได้เข้าใจโดยง่าย





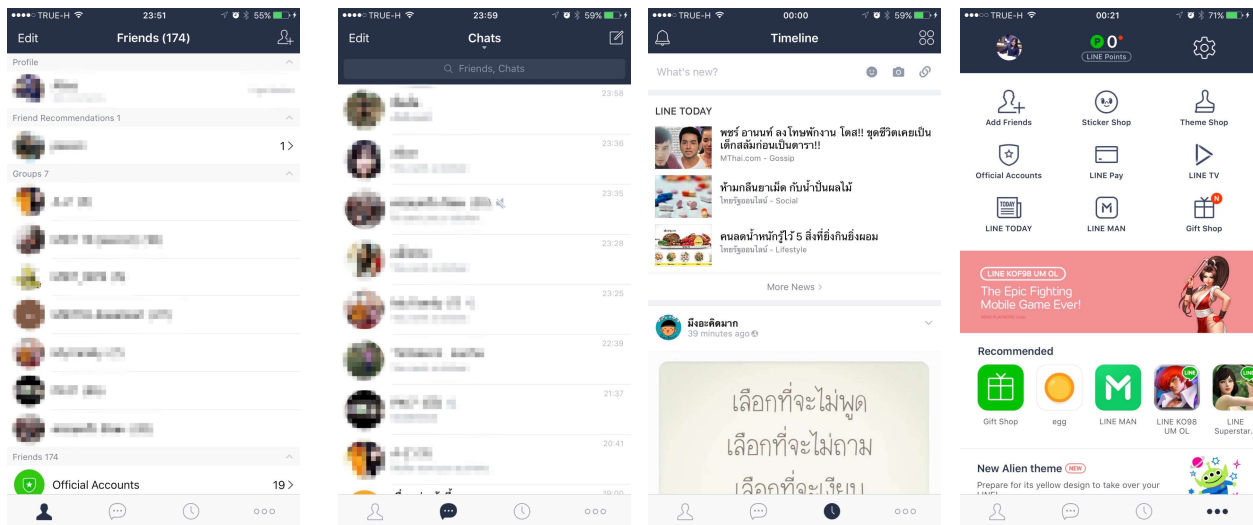
รายละเอียดโดยสรุปที่ทางLINEได้แสดงไว้มี 5 ข้อ คือ

1. Free Messaging : Make your chats come alive with an endless variety of stickers and themes.
= สามารถส่งข้อความได้ฟรี และสร้างควมมีชีวิตชีวาให้กับการสนทนาของเราด้วยการที่มีสติ๊กเกอร์ที่สามารถใช้ส่งหาคู่สนทนาได้และยังมีthemesที่ทำให้ห้องสนทนาของเรานั้นมีสีสันมากขึ้น
2. Free Voice & Video Calls : Talk with your LINE friends whenever you want for free!
= สามารถพูดคุยสนทนากับเพื่อนได้ฟรี ทั้งแบบสนทนาเฉพาะเสียง(voice call) และ แบบพูดคุยผ่านวิดีโอ(video call)
3. Stickers & Themes : Express yourself better with fun stickers and awesome theme packs!
= มีสติ๊กเกอร์ที่สามารถส่งหาเพื่อนๆได้ และมีธีมของหน้าการใช้งานที่สามารถเปลี่ยนได้ในแบบที่เราชอบ
4. Groups : Get closer to friends with groups
= เราสามารถสร้างกรุปที่ใช้สนทนากับเพื่อนๆได้
5. PC Version : Messaging has never been easier! Just link your smartphone LINE app with your computer.
= มีเวอร์ชันบนPCเพื่อเพิ่มความสะดวกสบายมากขึ้น โดยเราสามารถเชื่อมต่อเวอร์ชันที่อยู่บนPCของเรา กับเวอร์ชันที่อยู่บนสมาร์ทโฟนของเรา ทำให้เราสามารถสนทนากับเพื่อนได้โดยไม่สูญเสียความต่อเนื่อง

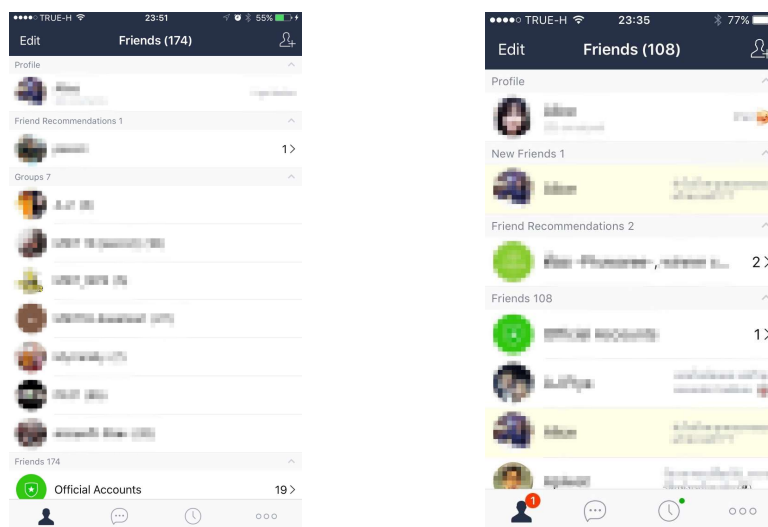
รูปแบบการใช้งาน

การใช้งานของ application line นั้น จะถูกแบ่งออกเป็นหน้าหลัก 4 หน้า คือ

1. หน้า Friends
2. หน้า Chats
3. หน้า Timeline
4. หน้า More



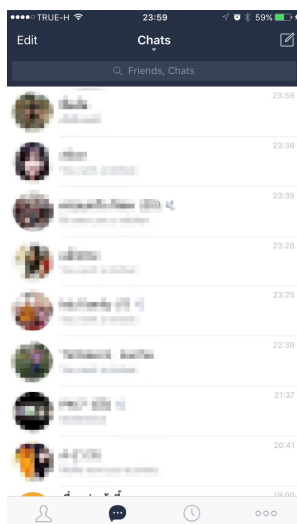
1. หน้า Friends



เป็นหน้าที่บอกว่าเพื่อนของเรามีใครบ้าง โดยจะมีการจัดแบ่งเป็น 5 ส่วนคือ

- 1.1. Profile = จะแสดงprofileของผู้ใช้งาน
- 1.2. New Friends = เพื่อนที่เพิ่งจะถูกเพิ่มเข้าไปในรายชื่อเพื่อน
- 1.3. Friend Recommendations = เพื่อนที่แนะนำ
- 1.4. Groups = กลุ่มของเพื่อนเราในlineนั้น เราสามารถจัดกลุ่มเพื่อนเพื่อคุยพร้อมกันที่ละหลายๆคนได้
- 1.5. Friend = รายชื่อบุคคลที่เป็นเพื่อนของเรา

2. หน้า Chats

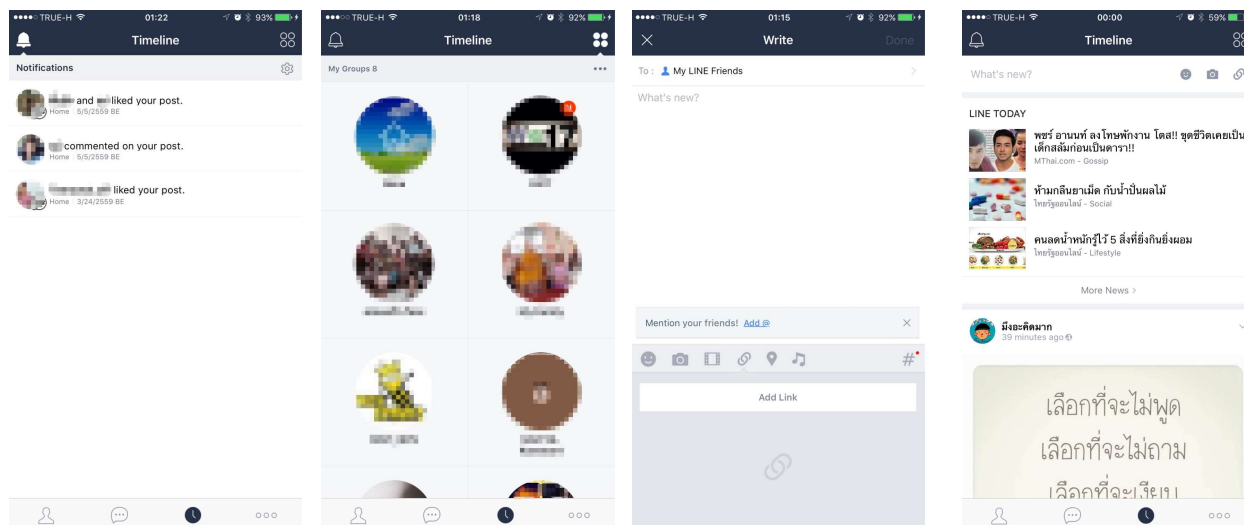


คือหน้าที่จะแสดงว่าเราคุยกับใครบ้าง ทั้งแบบรายคนและกลุ่ม ในส่วนของ Search นั้น เราสามารถ Searchหาได้ทั้งจาก ชื่อเพื่อน ชื่อกลุ่ม และ ข้อความที่คุยกันในห้องสนทนา

3. หน้า Timeline

เราจะมองเห็น 4 ส่วนหลักๆคือ

3.1. Notifications เมื่อเรากดไปที่รูปกระดิ่งที่อยู่ด้านบนทางซ้ายมือ หน้านี้จะถูกแสดงขึ้นมาเพื่อบอกเรา ว่า มีใครมาดlike หรือ commentเราบ้าง

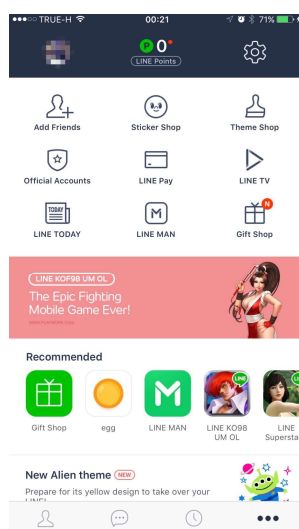


3.2. My Groups คือรูปวงกลม 4 วงที่อยู่ด้านบนทางขวามือ ในส่วนนี้จะแสดงว่าในห้องสนทนาของเรามีห้องไหนที่มีการโพสต์Notesอะไรบ้าง เพราะในห้องสนทนานั้นนอกจากการส่งข้อความโต้ตอบกันแล้ว เรายังสามารถโพสต์ ข้อความ รูปภาพ หรืออื่นๆ เป็นบอร์ดสนทนาเพื่อเติมได้อีกด้วย

3.3. Write (what's new?) เป็นส่วนที่เราจะสามารถโพสต์ข้อความ สติกเกอร์ รูปภาพ วิดีโอ URL location ให้แสดงในTimeline เพื่อให้เพื่อนของเราสามารถมาcomment กดlike กดshare ได้

3.4. Timeline ในส่วนนี้จะแสดง 2 ส่วน คือ LINE TODAY จะเป็นการรวบรวมข่าวสารจากเว็บไซต์ต่างๆ มาแสดงให้เราได้ดู และอีกส่วนคือ โพสต์ต่างๆที่เพื่อนของเราได้โพสต์ไว้

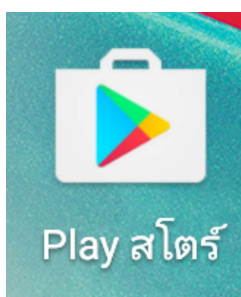
4. หน้า More

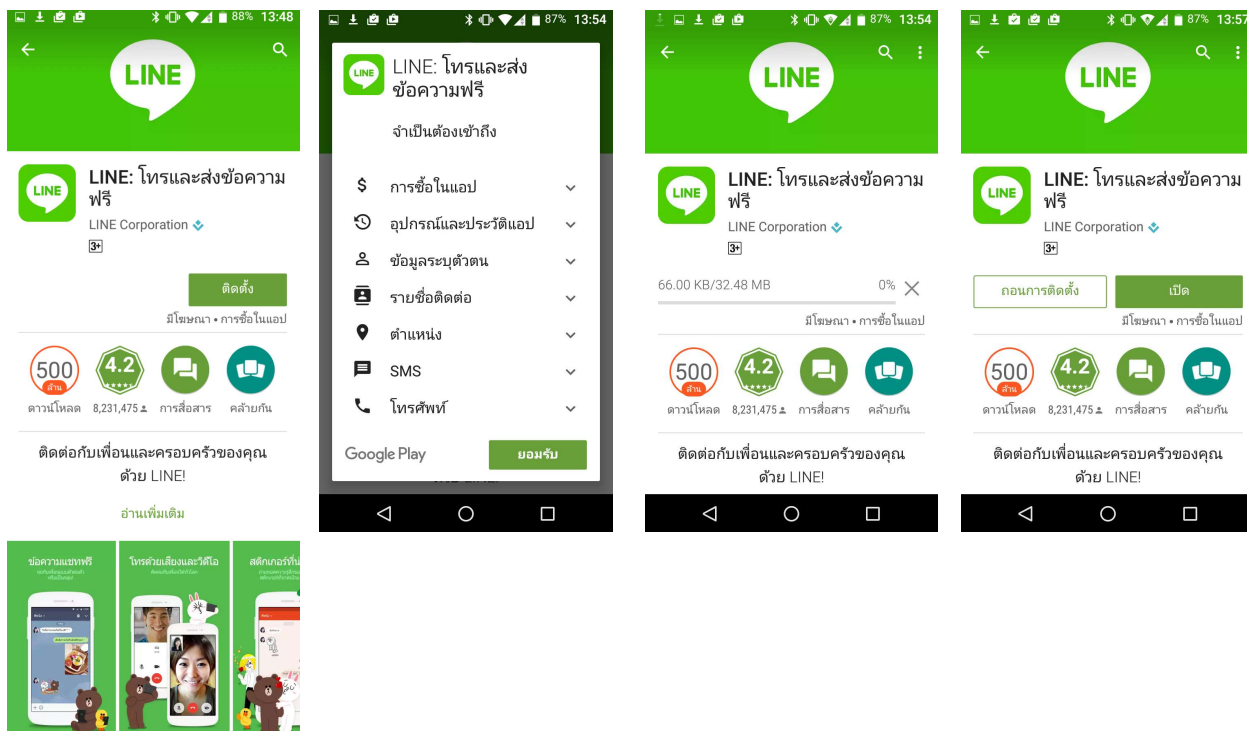


หน้านี้เป็นหน้าที่รวบรวมสิ่งต่างๆไว้ เช่น หน้าprofileของเรา , LINE Points , Settings , Add Friends , Sticker Shop , Theme Shop และ application หรือเกมอื่นๆที่อยู่ในเครือเดียวกันกับLINE เช่น LINE TV , LINE Camera , LINE Dictionary , LINE WEBTOON , LINE BUBBLE , LINE Rangers , Cookie Run

LINE บน Android

การใช้งาน LINE บน Android นั้น เริ่มต้นไม่ต่างอะไรจาก LINE บน ios มากนัก เราสามารถดาวน์โหลดแอปพลิเคชันได้จาก Play สโตร์

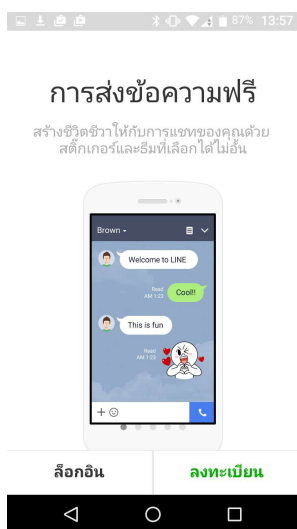




เมื่อเราเปิดเข้ามาในแอปพลิเคชันก็จะพบกับหน้าที่ให้เราเลือกอินหรือลงทะเบียน โดยหน้านั้นจะสรุปความสามารถของแอปพลิเคชันให้เราอย่างคร่าวๆมา 5 อย่าง เช่นเดียวกับบน ios คือ

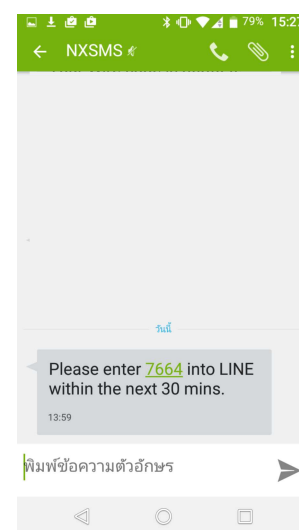
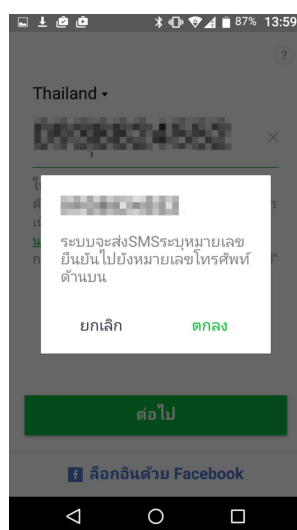
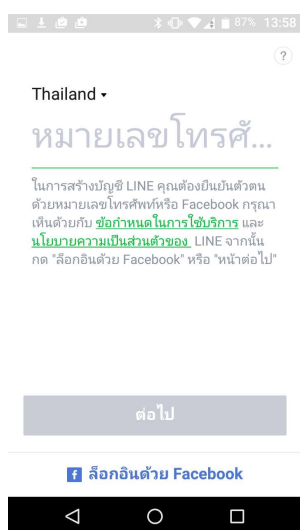
1. การส่งข้อความฟรี : สร้างชีวิตชีวาให้การแชทของคุณด้วยสติ๊กเกอร์และอิมที่เลือกได้ไม่อัน
2. โทรด้วยเสียงและวิดีโอฟรี : คุยกับเพื่อนในLINEได้ฟรีทุกเมื่อที่คุณต้องการ
3. สติ๊กเกอร์ที่สนุก : หลายฟังก์ชันของสติ๊กเกอร์และอิมที่สามารถแสดงความหลากหลาย
4. กลุ่ม : สนุกกับเพื่อนยิ่งขึ้นเมื่อมีกลุ่ม
5. เวอร์ชันพีซี : ส่งข้อความได้ง่ายกว่าที่เคย เพียงเชื่อมต่อแอปLINEบนสมาร์โฟนเข้ากับคอมพิวเตอร์

ของคุณ

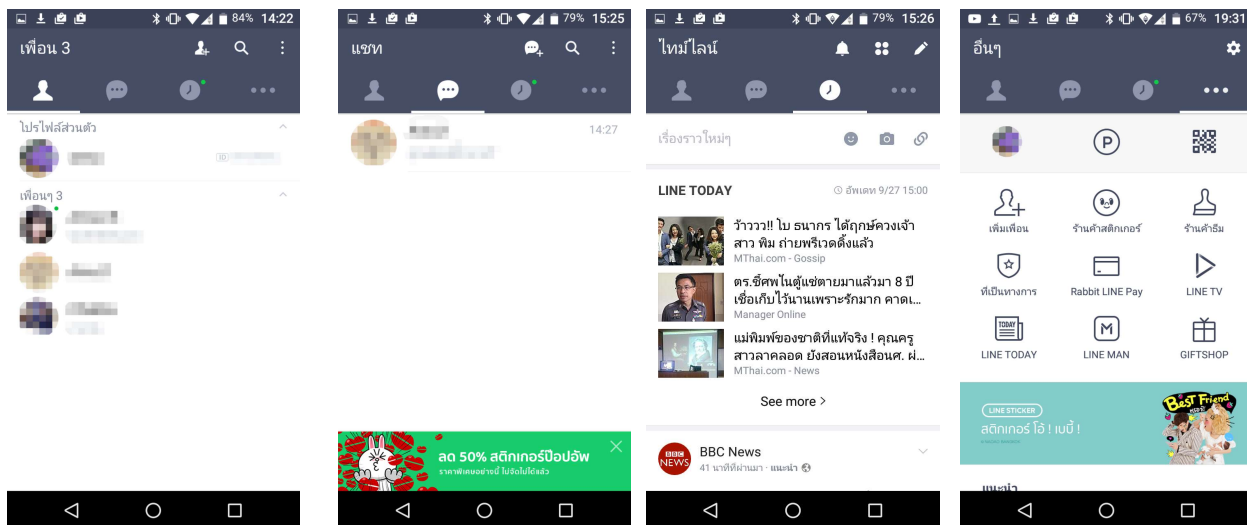




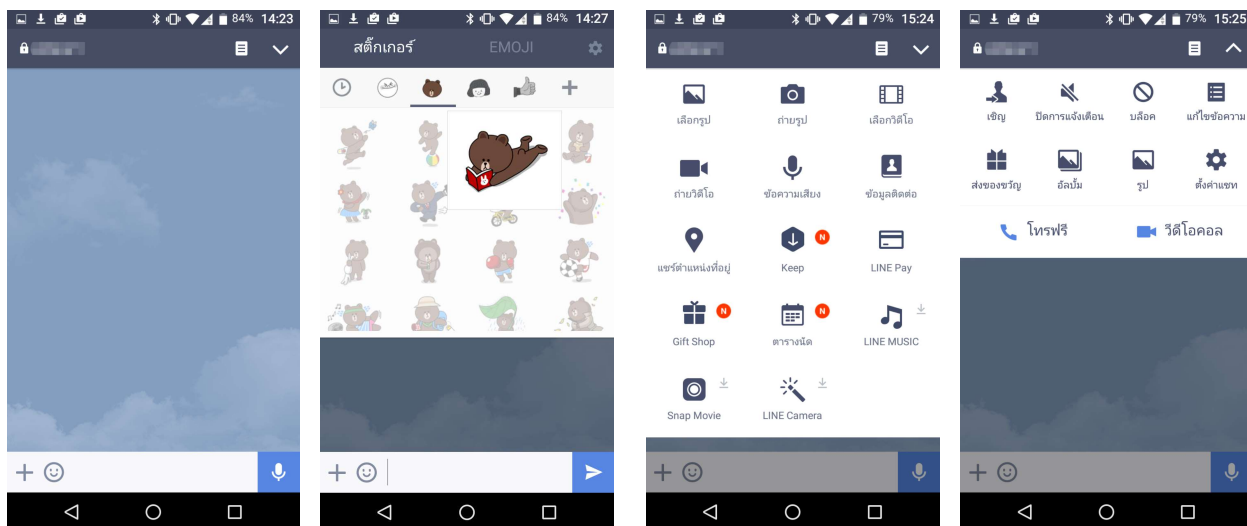
หากเรามีaccountของLINEอยู่แล้วเราก็สามารถล็อกอินได้เลย แต่ถ้าเราไม่มีก็ต้องทำการลงทะเบียนก่อน โดยแตะไปที่ลงทะเบียน หลังจากนั้นจะแสดงหน้าให้เราใส่หมายเลขโทรศัพท์ของเรา หลังจากนั้นเราใส่หมายเลขโทรศัพท์เสร็จแล้วระบบจะส่งsmsมาที่เครื่องของเราเพื่อเป็นการยืนยันตัวตน หลังจากนั้นก็สามารถใช้งานได้ทันที



เมื่อเราเข้ามาในหน้าการใช้งานแล้ว แอปพลิเคชันจะแสดงหน้าการใช้งานหลักๆ 4 หน้า ให้เราเลือกใช้งาน ซึ่งความสามารถทั้งหมดเหมือนกับLINEบนios แต่ที่แตกต่างคือ แลบที่ใช้ในการเลือกใช้งานหน้าต่างๆนั้นอยู่ด้านบน แต่บนiosนั้นอยู่ด้านล่าง และจะมีอีกแถบเครื่องมือ อีกแถบหนึ่งที่LINEบนiosไม่มีคือ แถบที่อยู่ด้านบนสุด ซึ่งใช้สำหรับการเพิ่มเพื่อน , ค้นหา , แก้ไขรายการเพื่อนและตั้งค่า



และเมื่อเราเข้ามาในหน้าchatsแล้ว เราก็จะพบว่าสิ่งที่แตกต่างกับเวอร์ชันของiosนั้นมีอีกอย่างคือ แถบเครื่องมือที่เราเรียกขึ้นมาจะอยู่ด้านบน ไม่ว่าจะป็นแถบของสติ๊กเกอร์หรือแถบของเครื่องมืออื่นๆ (สัญลักษณ์เป็นเครื่องหมายบวก) ซึ่งบนiosนั้นจะอยู่ด้านล่าง แต่แถบเครื่องมืออื่นๆหรือความสามารถต่างๆก็เหมือนกัน



Skype

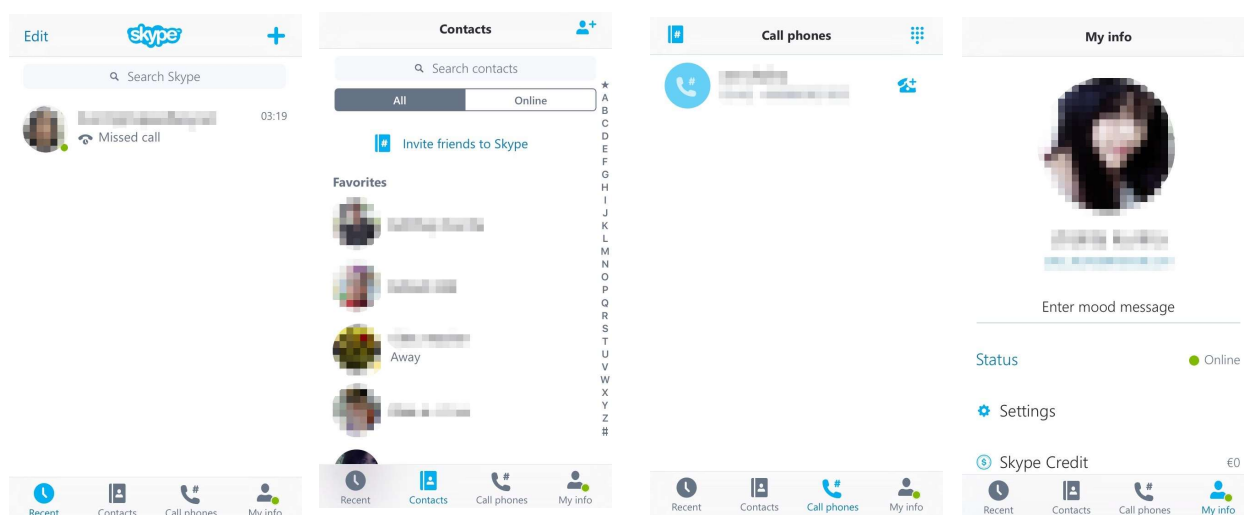


skype คือ บริการที่ทำให้สามารถข้อมูลทั้งทางเสียงพูด ภาพจริงขณะสนทนา การส่งข้อความ ส่งรูปภาพ ส่งวิดีโอ และการส่งข้อมูลในรูปแบบไฟล์ได้ โดยมีหน้าหลักในการใช้งานอยู่ 4 หน้า คือ Recent , Contacts , Call phones , My info

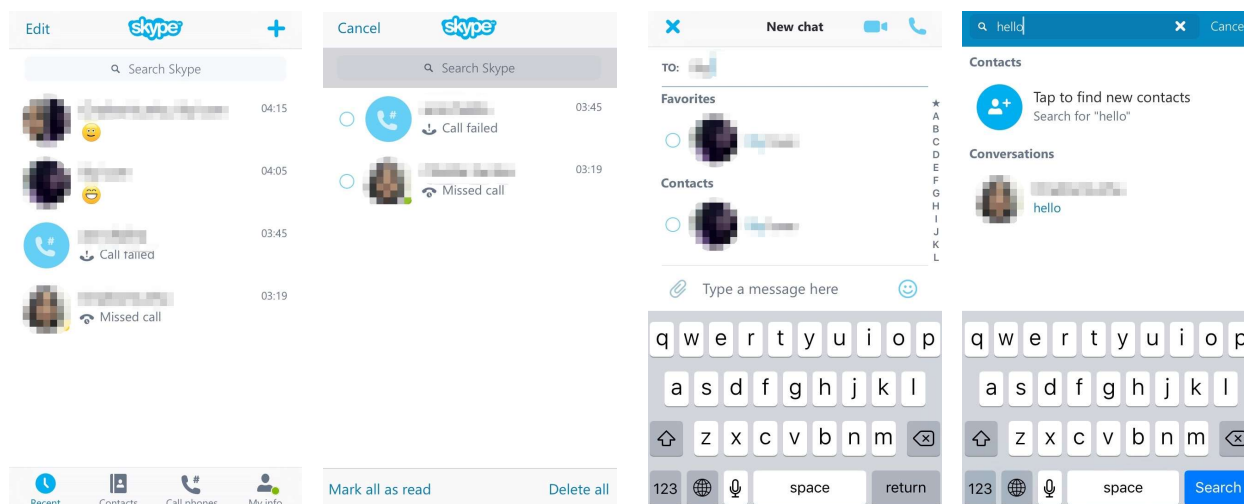
skype บน iOS

หลังจากที่เราดาวน์โหลดskypeมาจากApp Storeแล้ว เมื่อเราเปิดเข้าใช้งานในแอปพลิเคชันจะแสดงหน้าหลักๆที่เราสามารถใช้งานได้มาทั้งหมด 4 หน้า คือ

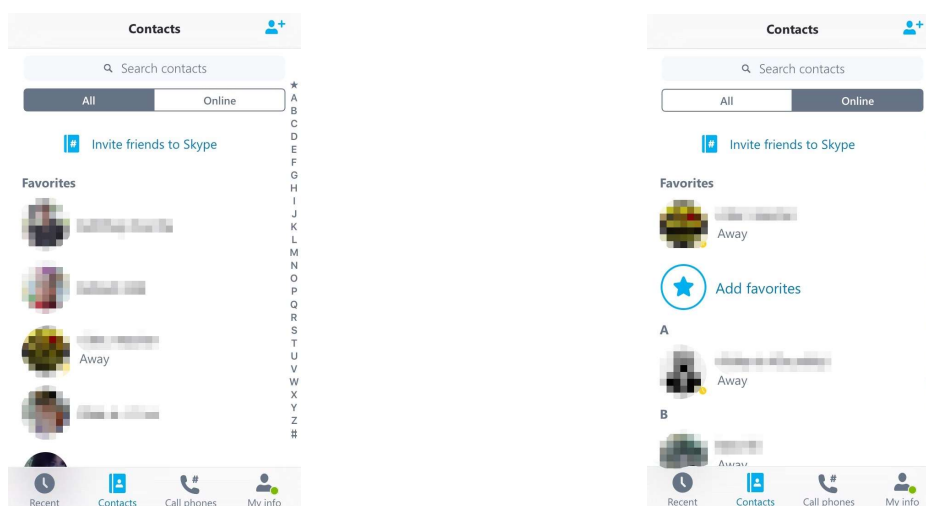
1. หน้า Recent
2. หน้า Contacts
3. หน้า call phones
4. หน้า My info



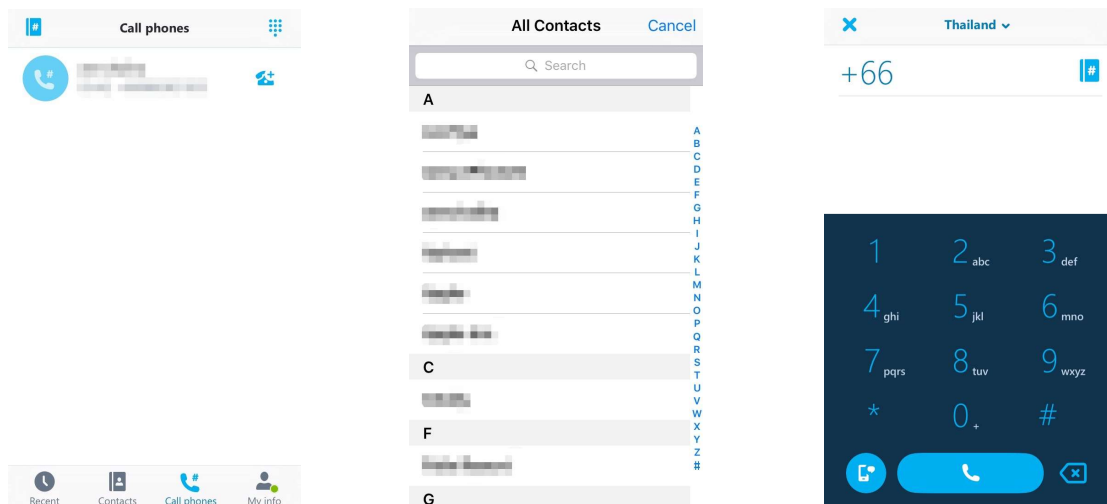
1. หน้า Recent เป็นหน้าที่บอกว่ามีใครส่งข้อความหาเราบ้าง คำว่าeditที่อยู่ทางซ้ายด้านบนทำให้สามารถลบห้องสนทนาได้ และสามารถเลือกได้ว่าจะลบเฉพาะห้องสนทนาที่เลือกไว้หรือจะลบห้องสนทนาทั้งหมด เครื่องหมายบวกที่อยู่ด้านบนทางขวาทำให้เราสามารถเลือกเพื่อนจากcontactsได้ว่าเราจะส่งข้อความหาใคร และเราสามารถคุยกับเพื่อนได้มากกว่าหนึ่งคนในหนึ่งห้องสนทนา ในส่วนของsearchสามารถค้นหาได้ทั้งชื่อเพื่อนในcontactsและหาโดยใช้คำค้นเพื่อหาว่าห้องสนทนาไหนมีคนที่เราใช้ในการsearchบ้าง



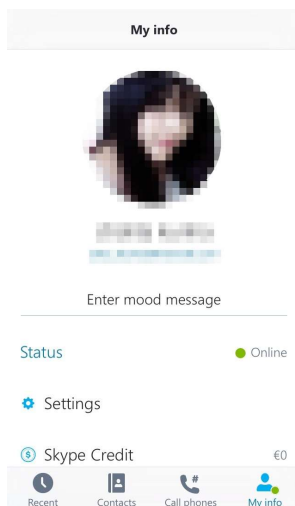
2. หน้า Contacts เป็นหน้าที่แสดงรายชื่อว่าเรามีใครเป็นเพื่อนบ้าง ซึ่งสามารถเลือกดูได้ว่า จะดูรายชื่อเพื่อนทั้งหมดหรือจะดูเฉพาะเพื่อนที่กำลังonline



3. หน้า call phones เป็นหน้าที่ให้เราสามารถโทรหาเพื่อนได้ โดยถ้ากดไอคอนด้านบนทางซ้ายจะแสดงรายชื่อในcontactsให้ แต่ถ้ากดไอคอนด้านบนทางขวาเราจะสามารถกดเบอร์โทรศัพท์ได้

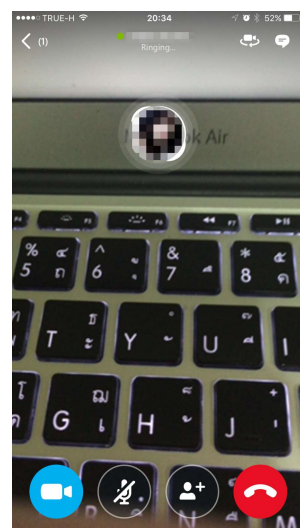
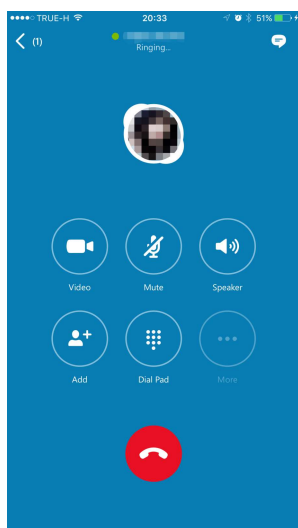
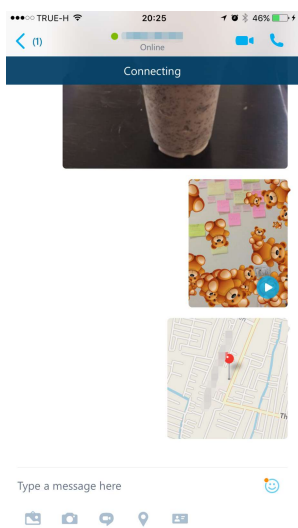
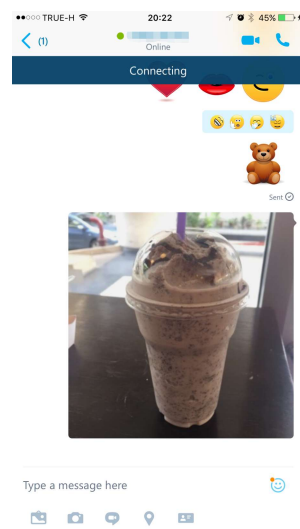
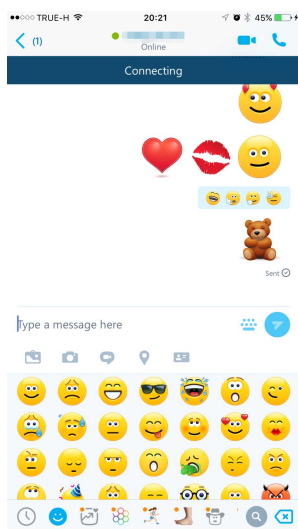
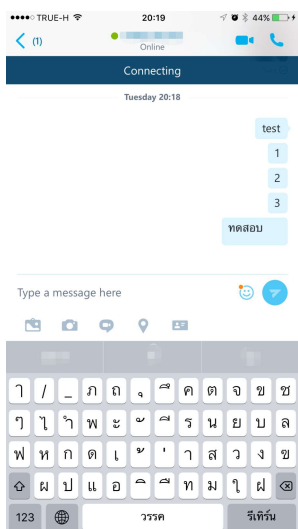


4. หน้า My info เป็นหน้าที่รวมข้อมูลต่างๆเกี่ยวกับaccountของเรา เช่น Status , Settings , Skype Credit , My account ฯลฯ



เมื่อพูดถึงความสามารถในการส่งข้อมูลหาเพื่อนนั้น สิ่งที่skypeทำได้อคือ

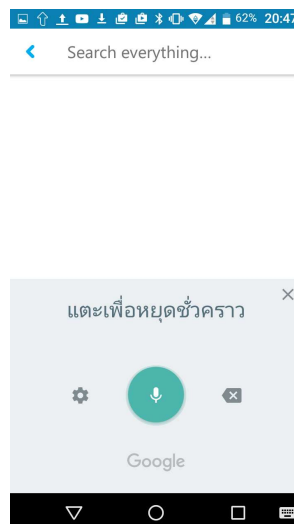
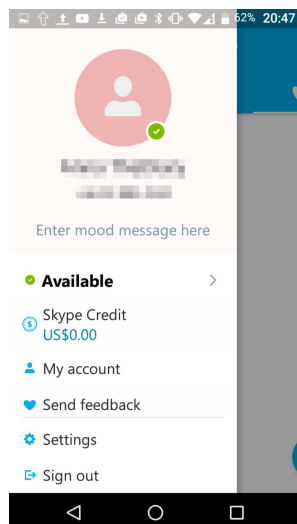
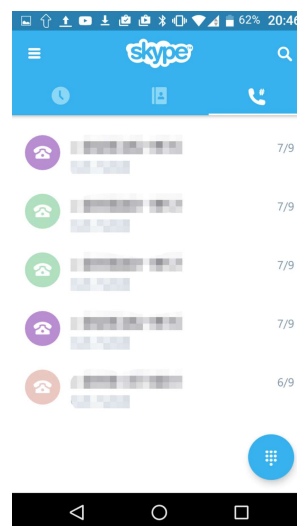
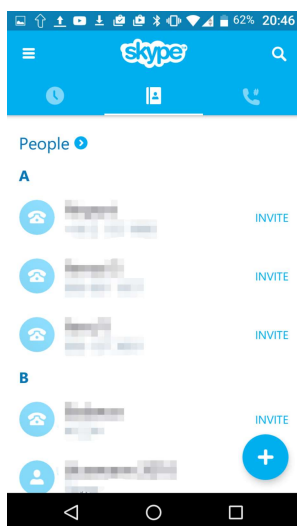
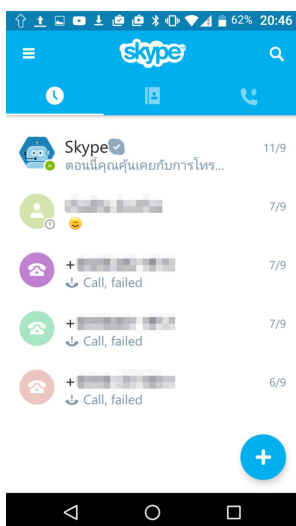
- สามารถส่งข้อความอักษรได้
- สามารถส่งข้อความภาพได้
- สามารถส่งข้อความวิดีโอได้
- สามารถส่งสติ๊กเกอร์หรืออีโมติคอนได้
- สามารถ share locationได้
- สามารถ share contactได้
- สามารถสนทนาได้ทั้งแบบเฉพาะเสียงและแบบวิดีโอ



Skype บน Android

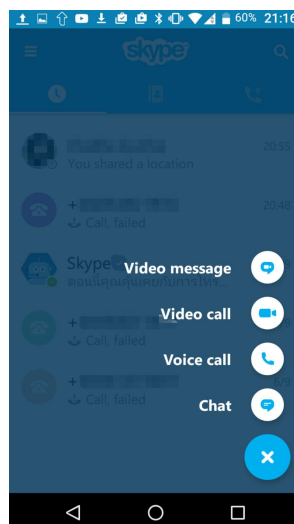
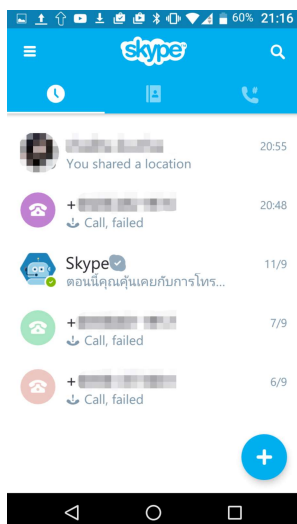
เมื่อเราเข้าใช้งานskypeบนAndroidแล้ว เราจะพบว่าหน้าตาของหน้าการใช้งานจะไม่เหมือนกับเวอร์ชันบนios แต่ก็ไม่ได้ต่างกันมาก โดยจะมีหน้าหลัก 5 หน้าให้เราเรียกใช้งาน คือ

1. Recent = เป็นหน้าที่บอกว่าเรามีการสนทนา หรือส่งข้อมูลหาเพื่อนคนไหน หรือเพื่อนคนไหนส่งข้อมูลมาหาเราบ้าง
2. Contacts = หน้านี้จะเป็นหน้าที่แสดงรายชื่อเพื่อนของเราจากสมุดรายชื่อในโทรศัพท์ (ส่วนนี้บนandroidจะต่างจากiosคือ บนandroidจะแสดงรายชื่อเพื่อนทั้งหมดในสมุดโทรศัพท์ของเครื่องโทรศัพท์ แต่บนiosจะแสดงแค่รายชื่อเพื่อนในskype)
3. Call phones = จะเป็นหน้าที่บอกว่าเรามีการโทรหาใครบ้าง
4. My info หรือ My account = จะบอกรายละเอียดของเรา
5. Search = จะเป็นหน้าค้นหา ซึ่งเราจะสามารถค้นหาได้ทั้งชื่อเพื่อน และข้อความที่เราคุยกับเพื่อน

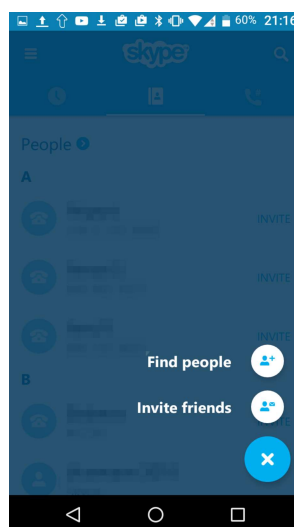
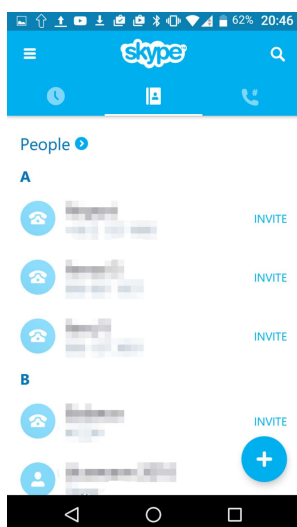


และเราจะสังเกตเห็นเครื่องหมายที่อยู่ด้านล่างขวาของจอ ซึ่งเครื่องหมายนี้มีความพิเศษคือ เมื่อเราเปิดหน้าแต่ละหน้า รายละเอียดที่แสดงขึ้นมาจะจะไม่เหมือนกัน โดยสามารถใช้งานได้และมีความแตกต่างกันทั้งหมด 3 หน้า

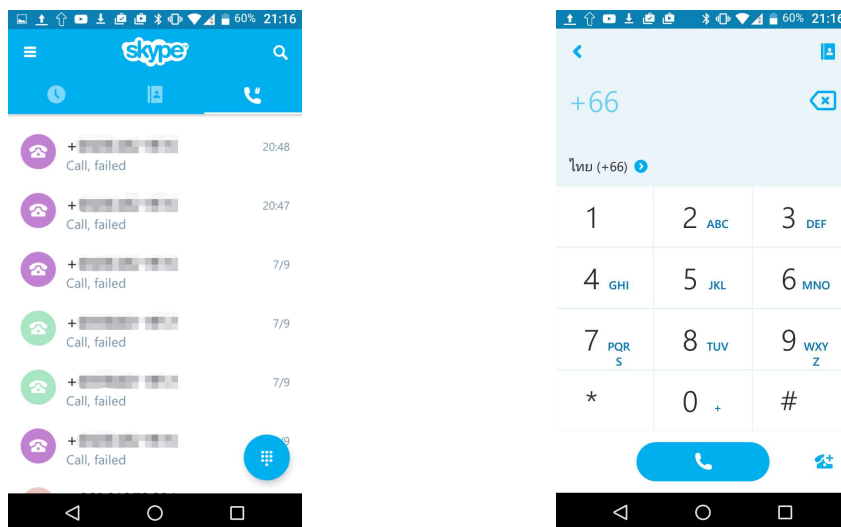
1. Recent = จะแสดงเป็นเครื่องหมายบวก โดยรายละเอียดด้านใน คือ Video message , Video call , Voice call , Chat



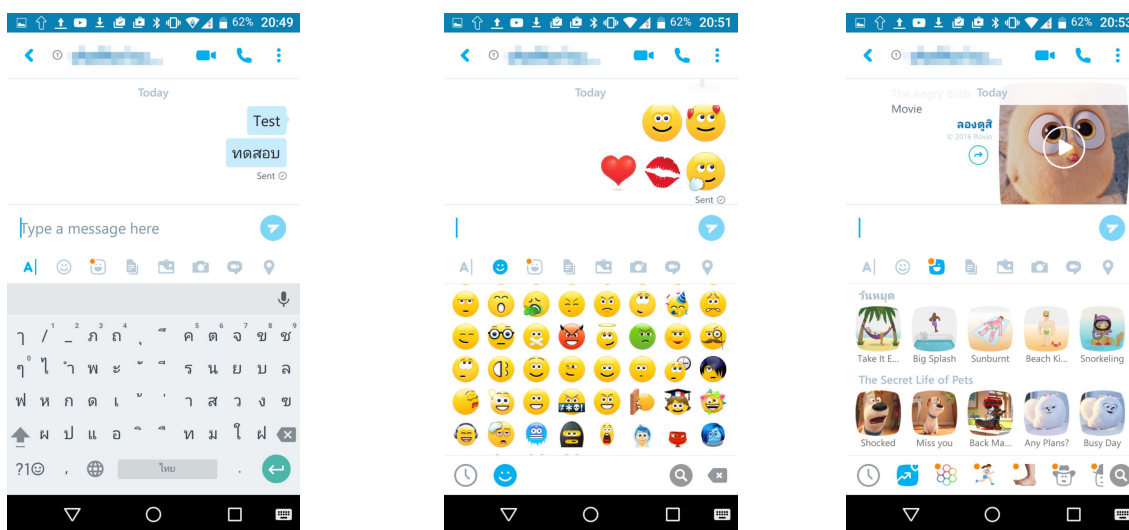
2. Contacts = จะแสดงเป็นเครื่องหมายบวกเช่นกัน รายละเอียดด้านใน คือ Find people , Invite friends

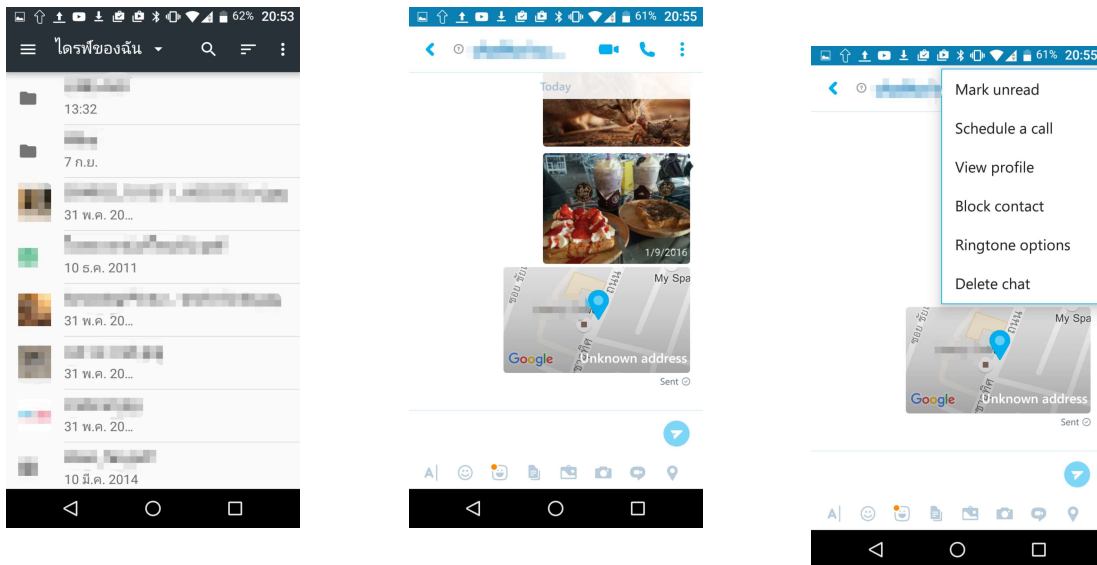


3. Call phone = หน้านี้เครื่องหมายที่แสดงจะไม่เหมือนกับหน้าอื่นๆ คือจะเป็นสัญลักษณ์จุดเล็กๆหลายจุด ด้านในวงกลม ซึ่งมาลักษณะคล้ายปุ่มกดโทรศัพท์ ซึ่งรายละเอียดด้านในก็จะแสดงหน้าจอของโทรศัพท์ขึ้นมาเพื่อให้เราสามารถโทรศัพท์หาเพื่อนได้โดยง่าย



และเมื่อเราเปิดหน้าห้องสนทนาขึ้นมา จะพบว่าห้องสนทนาของskypeบนandriodนั้น ถึงจะมีความคล้ายกับเวอร์ชันบนiosอยู่มาก แต่ที่เห็นได้ชัดที่สุดคือ แถบเครื่องมือด้านล่าง ที่จะแสดงเครื่องมือที่มากกว่าบนios ซึ่งที่มีเพิ่มเข้ามาจากiosก็คือ สามารถส่งไฟล์เอกสารได้ , มีสติ๊กเกอร์หรืออิมิติคอนที่เคลื่อนไหวและมีเสียงประกอบ และเรายังสามารถตั้งค่าต่างๆเพิ่มเติมได้จากเครื่องหมายจุด 3 จุด ด้านบนด้านขวา





แต่skypeบนandroidนั้นก็มีส่วนที่ในเวอร์ชันiosมี แต่androidไม่มีเหมือนกันนั่นก็คือ การshare contacts ซึ่งในiosมี แต่adroidไม่มี

การคุยแบบกรู๊ปนั้นskypeสามารถทำได้ทั้ง 2 os แต่ให้ความรู้สึกเหมือนเป็นกลุ่มชั่วคราวมากกว่า ที่จะ เป็นกลุ่มที่ตั้งกันแบบจริงจังถาวร(เมื่อเทียบกับแอปพลิเคชันอื่น)

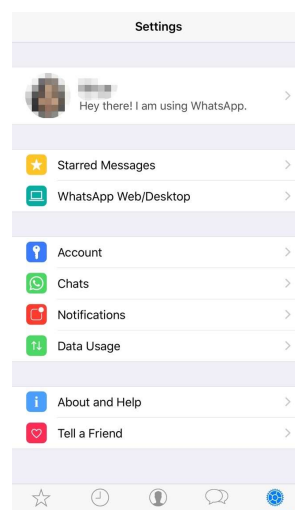
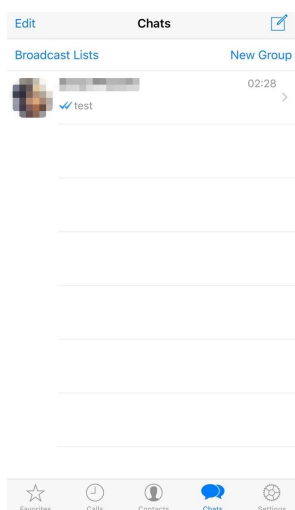
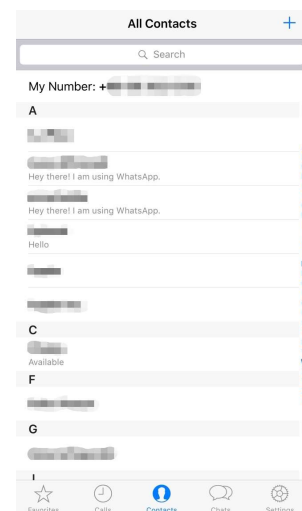
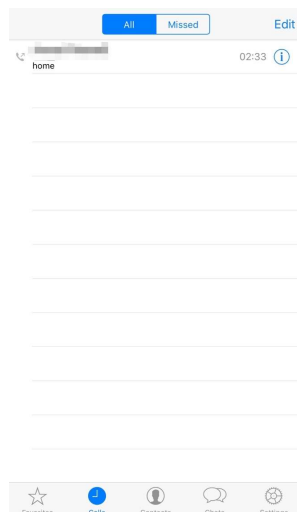
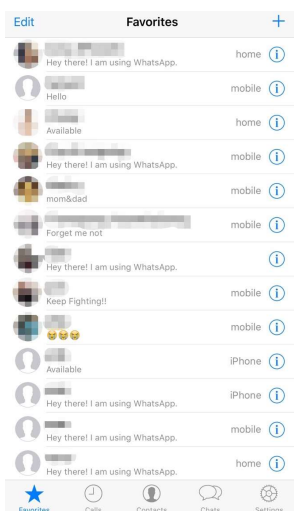
WhatsApp

WhatsApp คือ application ที่ทำให้สามารถส่งข้อมูลได้ทั้งบน ios , BB(Blackberry) , android โดยจะมีหน้าหลักอยู่ 5 หน้าคือ Favorites , Calls , Contacts , Chats , Settings

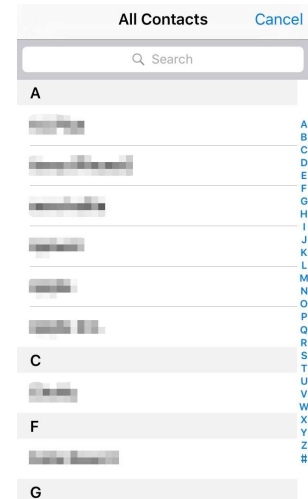
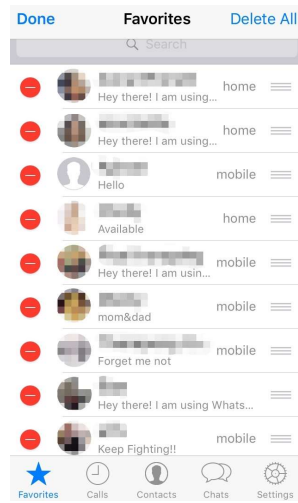
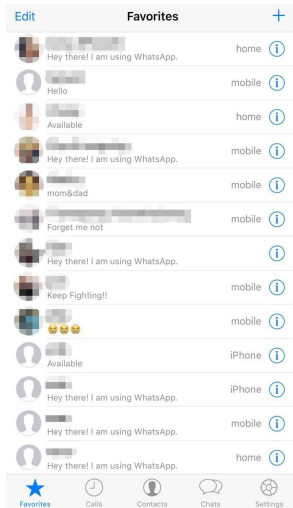
WhatsApp บน ios



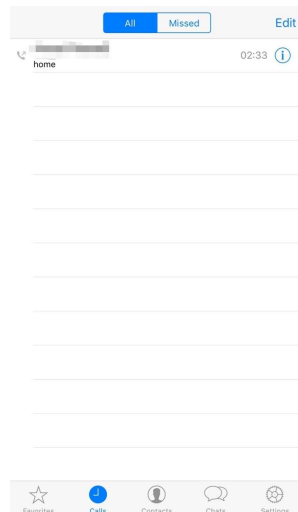
การใช้งาน WhatsApp บน ios นั้นจะมีหน้าหลักๆ ให้เราใช้งานได้ทั้งหมด 5 หน้า คือ Favorites , Calls , Contacts , Chats และ Settings



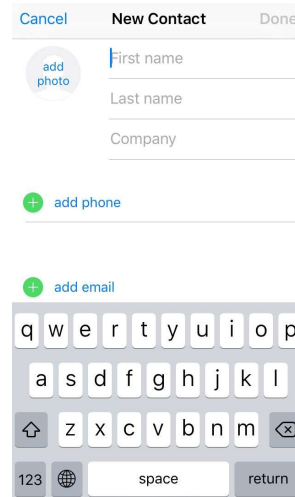
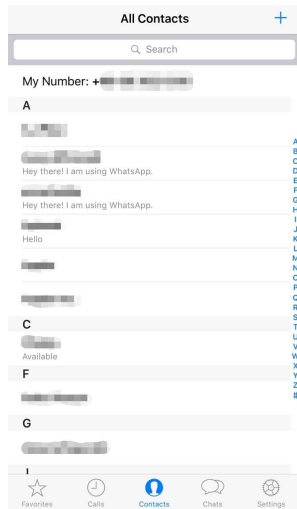
1. หน้า Favorites เป็นหน้าที่แสดงรายชื่อเพื่อน เราสามารถกด edit เพื่อลบรายชื่อเพื่อนหรือจัดลำดับก่อนหลังรายชื่อได้ หรือสามารถเพิ่มรายชื่อเพื่อนจาก Contacts ได้



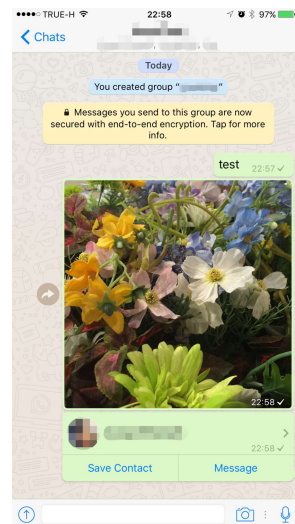
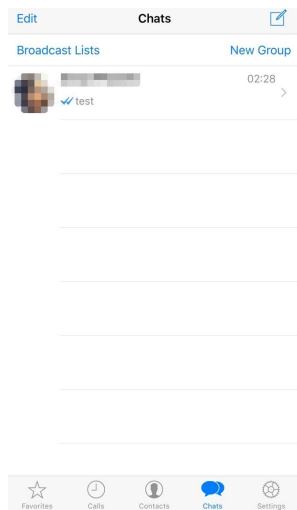
2. หน้า Calls จะแสดงว่าใครโทรหาเราหรือเราโทรหาใครบ้าง



3. หน้า Contacts เป็นหน้าที่แสดงรายชื่อจากContactsของเครื่องเรา และเราสามารถกดที่เครื่องหมายบวกที่อยู่ทางด้านบนทางขวาเพื่อเพิ่มรายชื่อเพื่อนได้

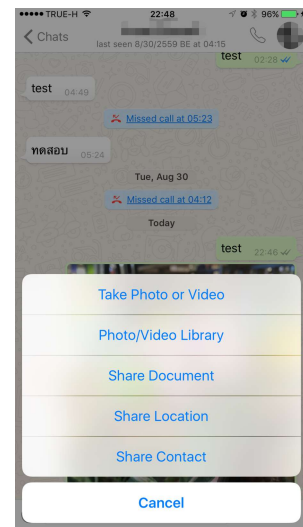
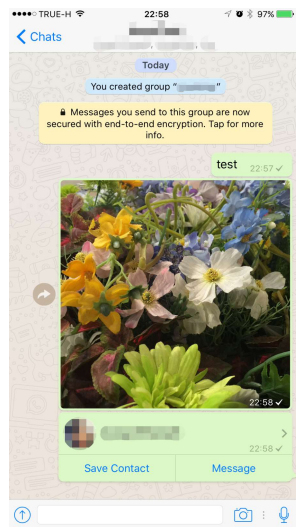
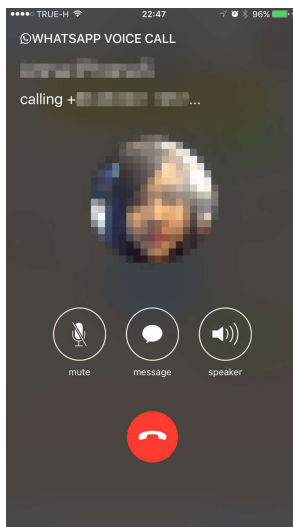


4. หน้า Chats หน้านี้จะแสดงรายชื่อเพื่อนที่เราส่งข้อความหา ซึ่งมีทั้งแบบรายบุคคล แบบBroadcast และแบบGroup

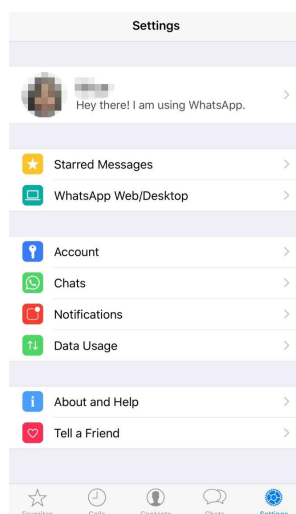


ซึ่งความสามารถในการส่งข้อมูลมีดังนี้

- สามารถสนทนาในรูปแบบเสียงได้ แต่ไม่สามารถสนทนาในรูปแบบวิดีโอได้
- สามารถส่งข้อความในรูปแบบอักษร และ เสียงได้
- สามารถส่งข้อความภาพ และ วิดีโอได้
- สามารถส่งไฟล์เอกสารได้
- สามารถ share location ได้
- สามารถ share contact ได้



5. หน้า Settings หน้านี้จะป็นหน้าสำหรับการตั้งค่าในส่วนต่างเช่น Profile , Account , Chats , Notification

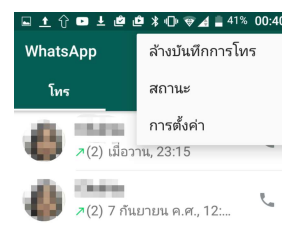
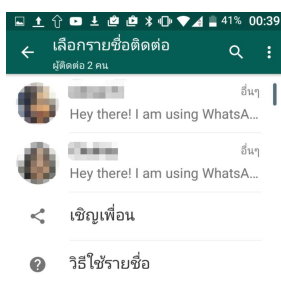
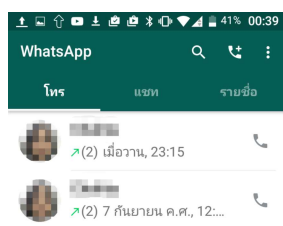


WhatsApp บน Android

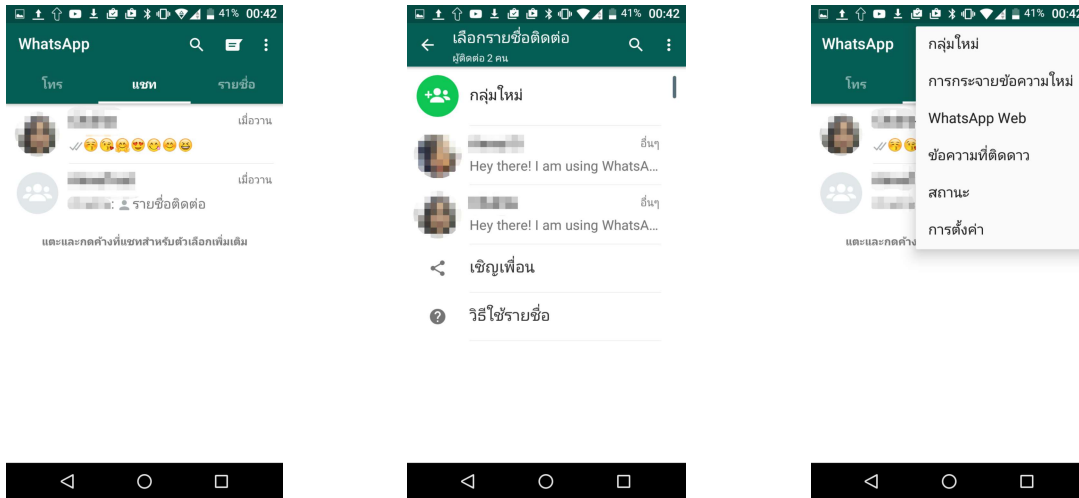


เมื่อเราเปิดเข้ามาในWhatsAppเวอร์ชันAndroidแล้ว เราจะพบว่าหน้าตาการใช้งานของแอปพลิเคชันนั้นไม่เหมือนกับเวอร์ชันiosเลย ในเวอร์ชันAndroidนี้ จะมีหน้าหลักทั้งหมด 3 หน้า และจะมีแถบเครื่องมืออยู่ด้านบนสุดด้านขวา โดยแถบเครื่องมือนี้จะเปลี่ยนไปตามหน้าการใช้งานที่เราเลือก

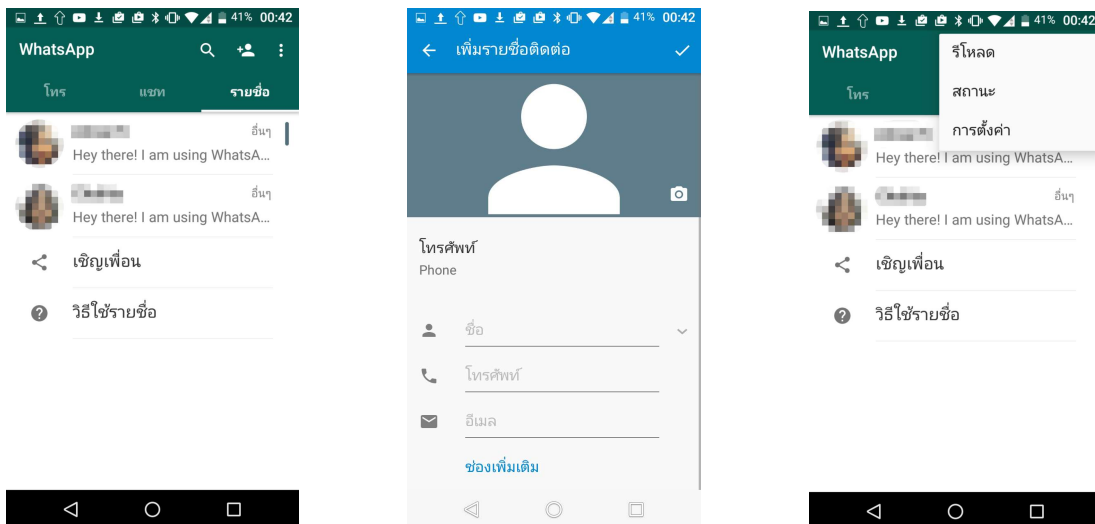
1. หน้าโทร : เป็นหน้าที่บอกว่าเราได้โทรหาเพื่อนคนไหนบ้าง ส่วนแถบเครื่องมือด้านบนนั้นจะประกอบไปด้วยเครื่องมือในการค้นหา , หน้าของการเลือกรายชื่อติดต่อ และสัญลักษณ์จุด 3 จุดก็จะประกอบไปด้วย ล้างบันทึกการโทร . สถานะ . การตั้งค่า



2. หน้าแชท : เป็นหน้าที่แสดงรายชื่อของเพื่อนที่เราได้มีการสนทนา ส่งข้อมูลต่างๆด้วย รวมถึงการสนทนาแบบกลุ่มด้วย และแถบเครื่องมือด้านบนก็มีส่วนของการค้นหา , รายชื่อติดต่อ , หน้าอื่นๆ ซึ่งด้านในประกอบไปด้วย กลุ่มใหม่ , การกระจายข้อความใหม่ , WhatsApp Web , ข้อความติดดาว , สถานะ , การตั้งค่า



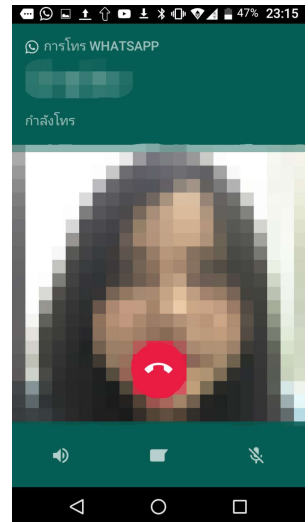
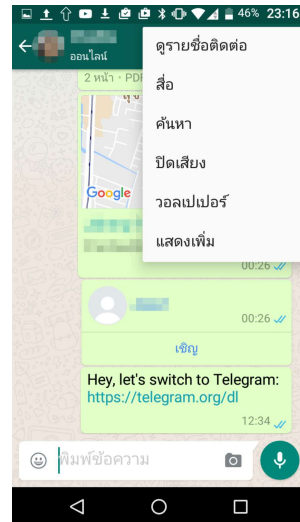
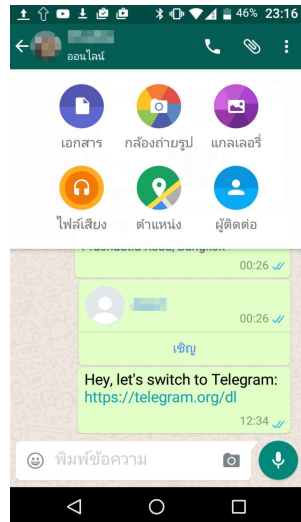
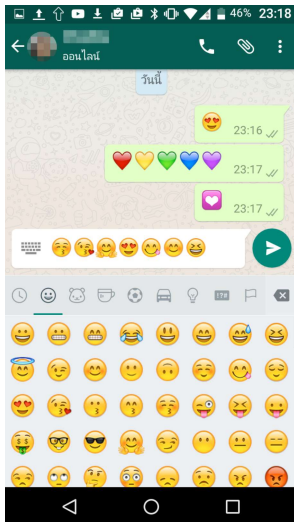
3. หน้ารายชื่อ : หน้านี้จะแสดงรายชื่อของเพื่อนที่ใช้งานskypeด้วยกัน แถบเครื่องมือในหน้านี้นอกจากจะมีหน้าค้นหาแล้ว ก็จะมีหน้าของการเพิ่มเพื่อน(เพิ่มรายชื่อติดต่อ) , หน้าอื่นๆก็จะประกอบไปด้วย รีโหลด , สถานะ และการตั้งค่า



ความสามารถในการส่งข้อมูลของแอปพลิเคชันนี้คือ

- สามารถส่งข้อมูลในรูปแบบอักษรและเสียงได้
- สามารถส่งข้อมูลในรูปแบบภาพและวิดีโอได้
- สามารถส่งข้อมูลในรูปแบบสติ๊กเกอร์หรือโมติคอนได้
- สามารถส่งไฟล์เอกสารได้

- สามารถโทรศัพท์หาเพื่อนได้(เฉพาะเสียง) แต่ไม่สามารถโทรศัพท์ในรูปแบบวิดีโอได้
- สามารถ share location ได้
- สามารถ share contact ได้



Telegram Messenger



Telegram เป็น application ที่ให้ความสำคัญกับความเร็วและความปลอดภัย โดยที่จุดเด่นของ application นี้คือ รวดเร็ว ง่าย ปลอดภัย และ ฟรี เราสามารถซิงค์ Telegram ได้กับทุกอุปกรณ์ คือ สามารถเล่นได้ทั้งบน smartphome และ desktop ใน Telegram สามารถเพิ่มเพื่อนได้อย่างไม่จำกัด สามารถสร้างกลุ่มเพื่อนได้ และในกลุ่มสามารถมีเพื่อนได้สูงสุดถึง 5,000 คน ข้อมูลที่ส่งสามารถส่งได้อย่างไม่จำกัด ส่งได้ทั้ง ข้อความ ภาพ เสียง และไฟล์ในรูปแบบต่างๆ เช่น doc , zip , pdf ฯลฯ

Telegram Messenger บน iOS

หลังจากที่เราดาวน์โหลดแอปพลิเคชันมาจาก App Store แล้ว เมื่อเริ่มใช้งานระบบจะให้เรายืนยันตัวตน โดยการใส่โค้ดที่ทาง Telegram จะส่งมาให้ใน smartphome ของเรา

◀ Back to App Store 01:02 100%



Telegram

The world's fastest messaging app.
It is free and secure.

Start Messaging >



Code

We have sent you an SMS with the code

Telegram will call you in 1:34



TRUE-H 07:21 92%

< Messages (13) Verify Details

Text Message
23 ก.ค. 2557 09:19

ใช้ 00285 เป็นรหัสความปลอดภัยบัญชี Microsoft

พ. 27 ก.ค. 09:16

Telegram code 48227

Telegram code 85467

Telegram code 45699

Telegram code 58424

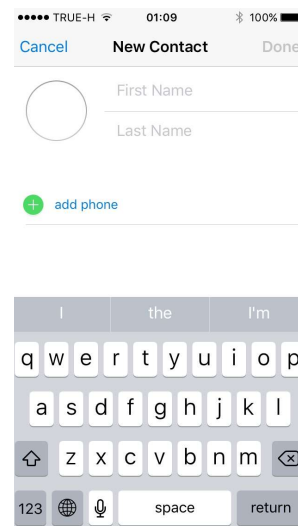
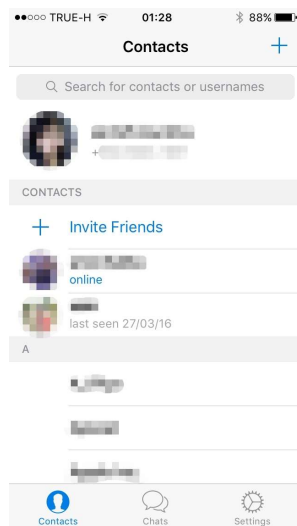
วันนี้ 01:03

Telegram code 93607

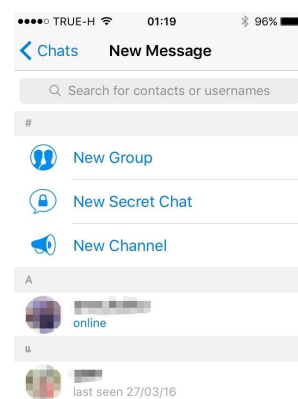
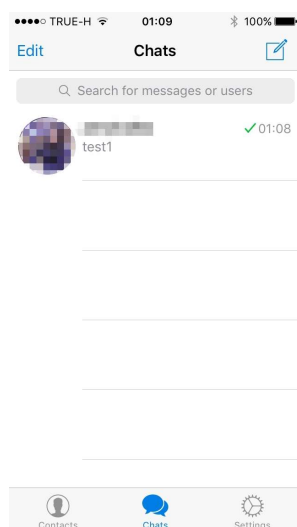
Text Message Send

เมื่อเปิดเข้าไปในapplicationจะถูกแบ่งเป็น 3 หน้าหลักๆคือ หน้าของ Contacts , Chats , Setting

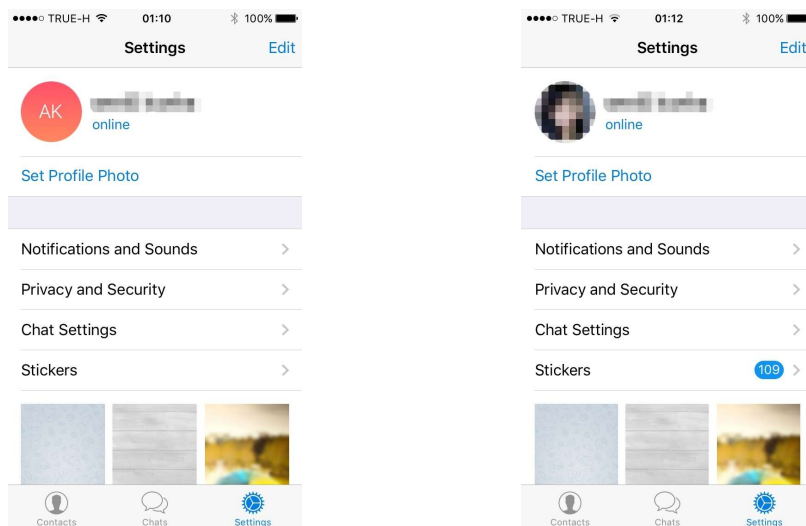
1. Contacts เมื่อเราเปิดเข้าไปในหน้านี้จะมองเห็นด้านบนสุดคือaccountของเรา ถัดลงมาจะเป็นรายชื่อของเพื่อนที่อยู่ในTelegram สุดท้ายจะเป็นรายชื่อของเพื่อนที่อยู่ในสมุดโทรศัพท์ในเครื่องของเรา ซึ่งเราสามารถ invite เพื่อน และสร้าง contact ใหม่ของเพื่อนเราได้



2. Chats หน้าที่จะแสดงรายชื่อเพื่อนที่เราได้มีการส่งข้อมูลหา ด้านบนสุดเราสามารถกดเข้าไปเพื่อสร้างกลุ่มเพื่อนเพื่อส่งข้อมูลได้

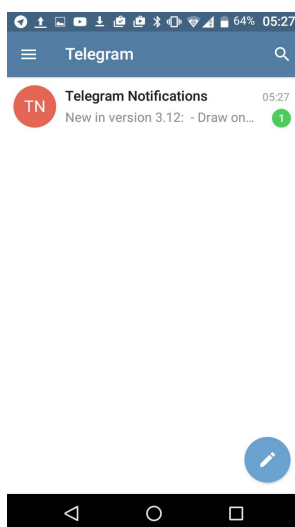


3. Settings จะเป็นส่วนที่เราจะสามารถเข้ามาตั้งค่าต่างๆได้ เช่น รูปprofile , การเตือนต่างๆ , การตั้งค่าการสนทนา , การติดตั้งสติ๊กเกอร์ , การตั้งภาพเบื้องหลัง ฯลฯ

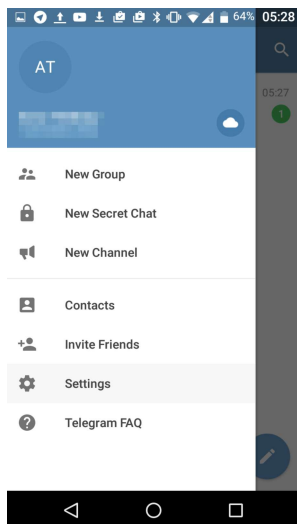


Telegram Messenger บน Android

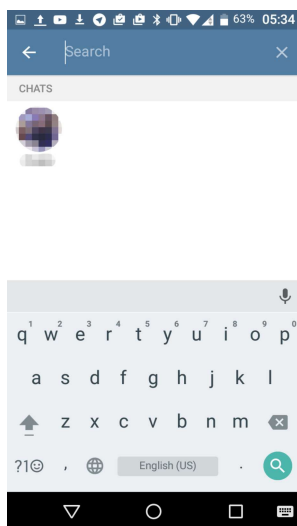
เมื่อเราเปิดเข้าไปใน Telegram Messenger เวอร์ชันบน Android เราจะพบว่าหน้าต่างของแอปพลิเคชันเมื่อเปรียบเทียบกับเวอร์ชันบนiosแล้วมีความแตกต่างกันอย่างมาก เวอร์ชันAndroidนั้นเมื่อเราเข้ามาในหน้าแรกนั้น หน้านี้จะเป็นหน้าที่แสดงให้เราเห็นว่าเรามีการติดต่อกับใครบ้าง และในหน้านี้จะป็นหน้ากลางให้เราไปสู่หน้าอีก 3 หน้า คือ หน้าAccount , หน้าค้นหา , หน้าNew Message



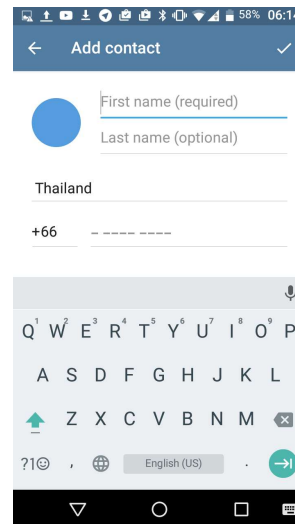
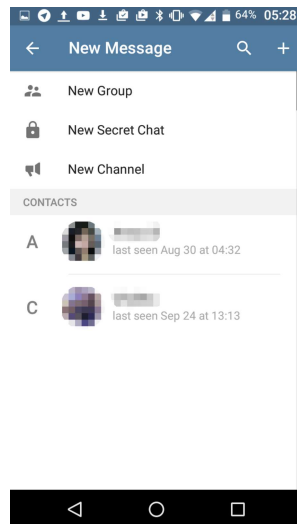
หน้าAccount หน้านี้จะจะเป็นหน้าที่รวบรวมการใช้งานหลายๆอย่างไว้ เช่น การสร้างกรุป , การเปิดใช้งานSecret Chat , การสร้างChannel , Contacts , invite Friends , Settings , การรวบรวมคำถามต่างๆที่มีต่อTelegram



หน้าค้นหา หน้านี้เราจะสามารถใช้ค้นหาเพื่อนที่เราต้องการจะติดต่อด้วย โดยคำค้นนั้นสามารถค้นหาได้ทั้งจากชื่อเพื่อนและข้อความที่ใช้ในห้องสนทนา

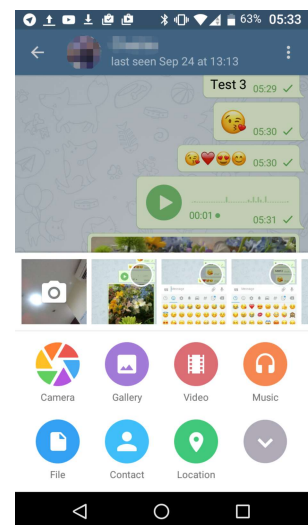
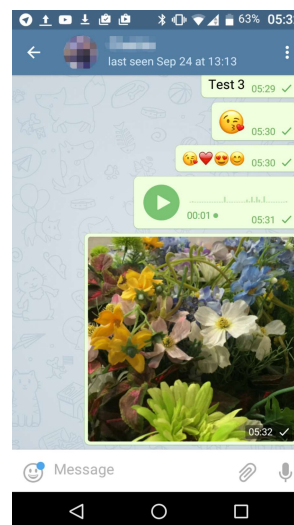
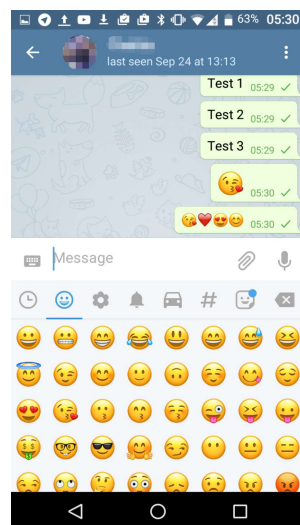
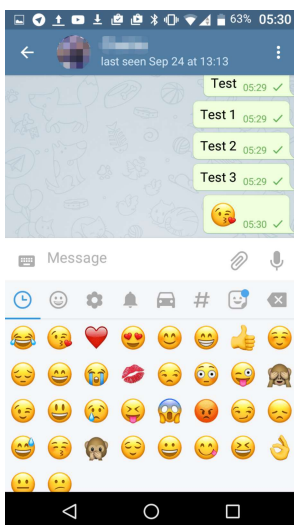


หน้าNew Message หน้านี้เราสามารถสร้างกรุปได้ , สามารถสร้างห้องSecret Chat , สร้างChannelใหม่ได้ , สามารถเพิ่มcontactได้ และcontactsของเราก็จะแสดงอยู่ในหน้านี้ด้วย

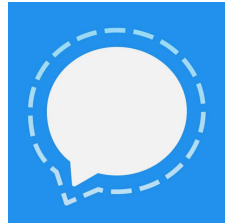


ในหน้าของห้องสนทนานั้นเราจะพบกับความสามารถต่างๆในการส่งข้อมูลของTelegramคือ

- สามารถส่งข้อมูลในรูปแบบอักษรและเสียงได้
- สามารถส่งข้อมูลในรูปแบบภาพและวิดีโอได้
- สามารถส่งข้อมูลในรูปแบบสติ๊กเกอร์หรืออีโมติคอนได้
- สามารถส่งข้อมูลในรูปแบบไฟล์เอกสารได้
- สามารถส่งเพลงให้เพื่อนได้
- สามารถ share locationได้
- สามารถ share contactsได้
- ไม่สามารถโทรศัพท์หาเพื่อนได้ทั้งในรูปแบบเสียงและวิดีโอ



Signal



logoบนiOS

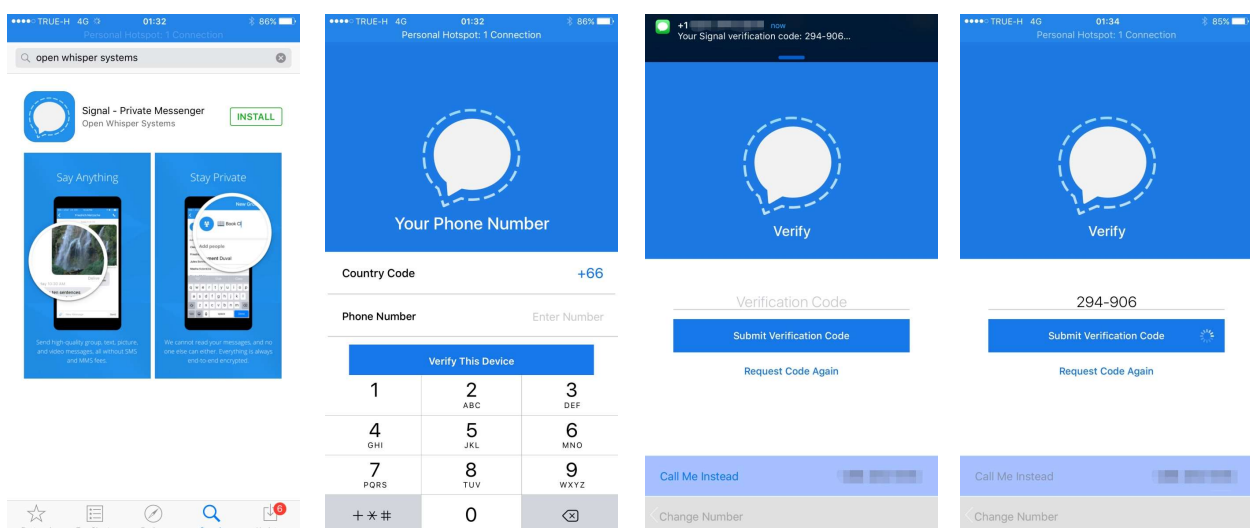


logoบนandroid

ความแตกต่างที่เห็นได้อย่างชัดเจนของ Signal ระหว่าง iOS และ android คือ logo ของ signal ส่วนเรื่องของความสามารถนั้นก็มีความแตกต่างกันเล็กน้อย เช่น บน ios จะไม่สามารถส่งสติ๊กเกอร์หรือไอคอนรูปภาพได้ , บน ios ไม่สามารถ share location ได้ แต่บน android สามารถshare locationได้ , บน ios ไม่สามารถ share contact ให้เพื่อนได้ แต่บน android สามารถทำได้ และ บน ios เราจะสามารถcapหน้าจอได้ แต่บน androidไม่สามารถทำได้ เนื่องจากติด DRM (Digital Rights Management)

Signal บน iOS

หลังจากที่เราได้ทำการติดตั้ง Signal บนiosแล้ว จะโชว์หน้าของการลงทะเบียนใช้งานโดยจะให้เราได้หมายเลขโทรศัพท์ของเราและจะมีการส่ง sms เพื่อยืนยันตัวตนเราด้วย

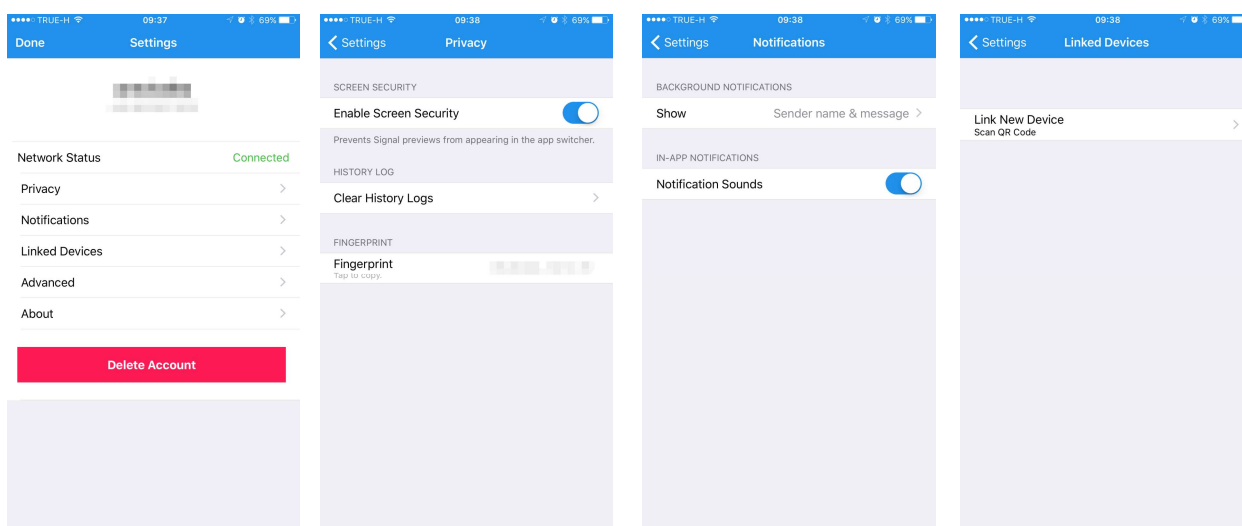


เมื่อเราลงทะเบียนและผ่านการยืนยันตัวตนเรียบร้อยแล้ว เราจะเข้ามาที่หน้าของการใช้งาน ซึ่งเริ่มแรกจะยังเปิดหน้าเปล่าอยู่เพราะเรายังไม่ได้มีการส่งข้อมูลหาใคร เมื่อได้มีการส่งข้อมูลหาเพื่อนเราแล้วชื่อของเพื่อนเราจะมาแสดงในหน้านี้ การส่งข้อมูลหาเพื่อนนั้นเราสามารถส่งได้จากการแตะที่รูปดินสอที่มุมขวาบนของจอ จะแสดงรายชื่อของเพื่อนให้เราเลือกที่เราจะส่งข้อมูลหาเพื่อนคนไหน ส่วนมุมซ้ายของจอจะเป็นส่วนของหน้าการตั้งค่า



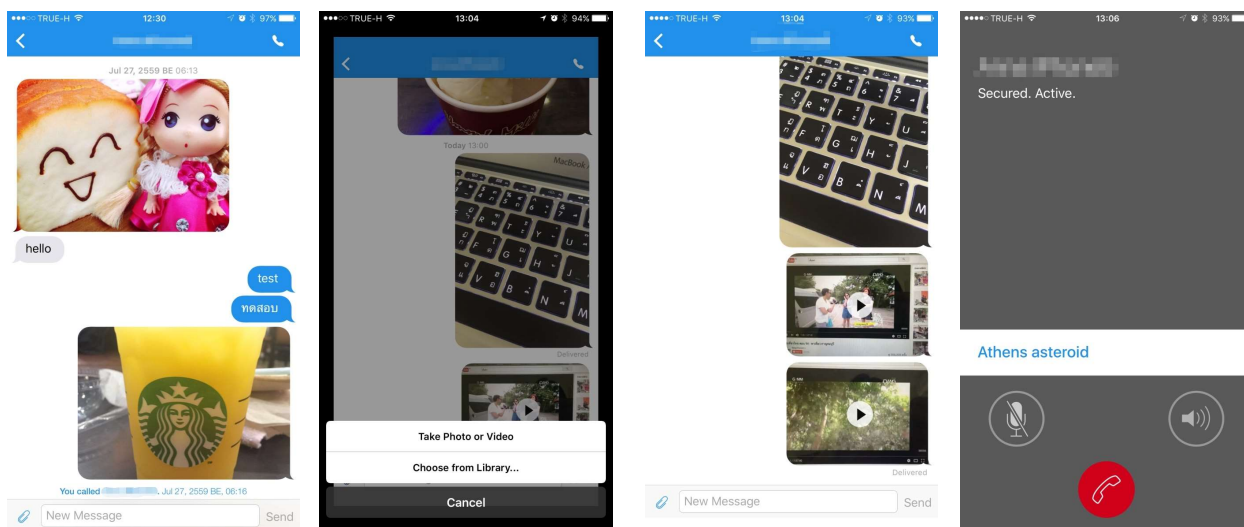
Start your first Signal conversation!
Tap on the compose button.

ในหน้าการตั้งค่านั้นก็จะมีให้เราเลือกที่เราจะตั้งค่าอะไรบ้าง หรือในบางส่วนก็จะแสดงข้อมูลเกี่ยวกับ account ของเรา เช่น Privacy , Notification , Linked Devices



ในหน้าสนทนานั้นSignalมีความสามารถต่างๆดังนี้

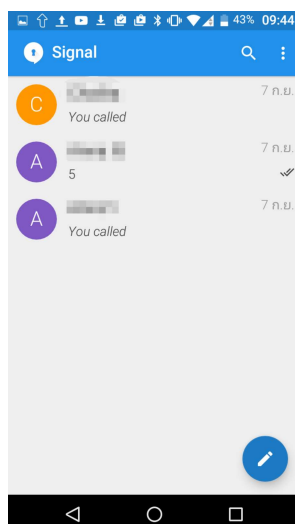
- สามารถส่งข้อมูลในรูปแบบอักษร แต่ไม่สามารถส่งข้อมูลในรูปแบบเสียงได้
- สามารถส่งข้อมูลในรูปแบบภาพและวิดีโอได้
- สามารถส่งข้อมูลในรูปแบบสติ๊กเกอร์หรืออีโมติคอนได้
- ไม่สามารถส่งข้อมูลในรูปแบบไฟล์เอกสารได้
- สามารถสนทนาในรูปแบบเสียงได้ แต่ไม่สามารถสนทนาในรูปแบบวิดีโอได้
- ไม่สามารถ share location หรือ share contacts ได้
- สามารถสร้างกลุ่มเพื่อนเพื่อส่งข้อมูลหาเพื่อนเป็นกลุ่มได้



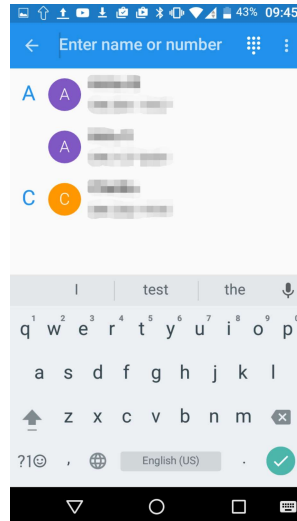
Signal บน Android

เมื่อเราเปิดเข้ามาใน Signal บน Androidแล้ว เราจะเห็นว่ามีความแตกต่างกันเล็กน้อยกับ Signal บน iOS เวอร์ชันบนandroidนั้นมีส่วนที่สามารถใช้งานได้อยู่ทั้งหมด 3 ส่วนหลักๆ คือ

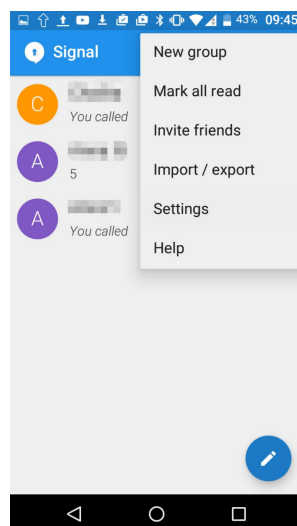
1. หน้าแรก หรือ หน้ากลาง ซึ่งจะเป็นหน้าที่เราจะพบเป็นหน้าแรกในการเปิดเข้าใช้งานSignalเลย ในหน้านี้จะแสดงรายชื่อเพื่อนที่เราได้มีการติดต่อส่งข้อมูลหากันไว้ และเป็นหน้ากลางที่จะเชื่อมไปหาหน้าในการใช้งานต่างๆ



2. หน้าค้นหา จะมี 2 ส่วนคือ รูปแว่นขยายด้านบนขวามือ และ รูปดินสอที่อยู่ด้านล่างขวามือ ทั้ง 2 หน้านี้จะทำหน้าที่เหมือนกันคือ สามารถหาชื่อเพื่อนได้จากรายชื่อของเพื่อนและเบอร์โทรศัพท์ ซึ่งเราสามารถพิมพ์ลงไปหรือจะพูดเพื่อให้แอปพลิเคชันพิมพ์ให้ก็ได้

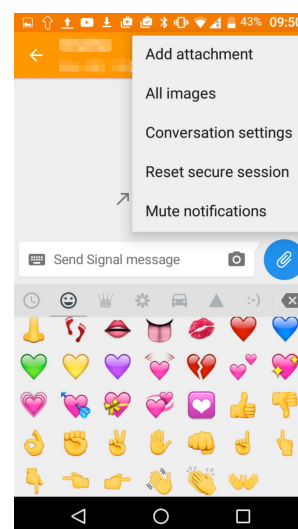
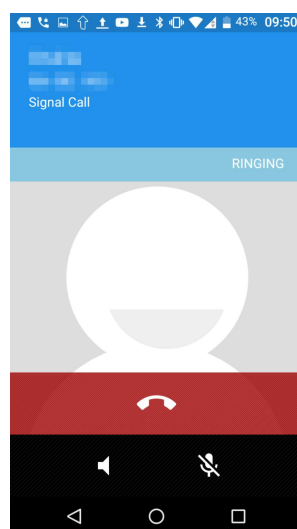
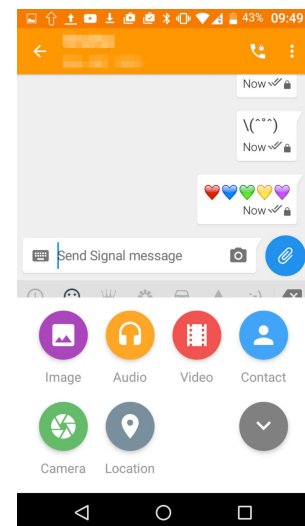
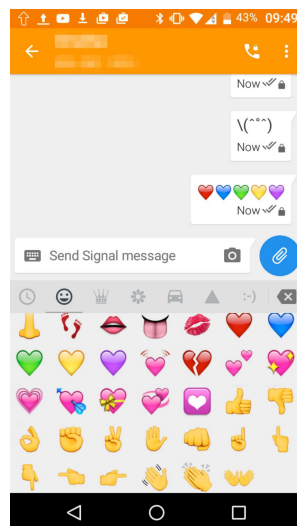
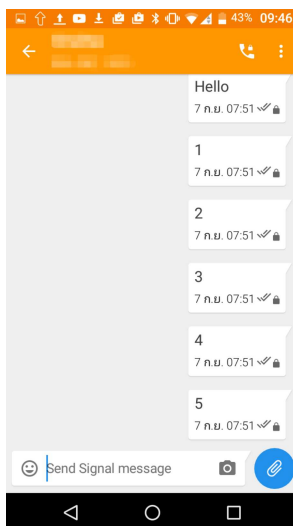


3. หน้า เครื่องมืออื่นๆ ซึ่งจะมีสัญลักษณ์เป็นจุด 3 จุด อยู่บนสุดขวามือ ในหน้านี้นี้จะประกอบด้วย New group , Mark all read , Invite friends , Import / export , Settings , Help



ในหน้าสนทนาของSignal บน Androidนั้น เราสามารถส่งข้อมูลต่างได้ดังนี้

- สามารถส่งข้อมูลได้ทั้งในรูปแบบอักษร และรูปแบบเสียง
- สามารถส่งข้อมูลในรูปแบบภาพและวิดีโอได้
- สามารถส่งข้อมูลในรูปแบบสติ๊กเกอร์หรืออีโมติคอนได้
- ไม่สามารถส่งข้อมูลในรูปแบบไฟล์เอกสารได้
- สามารถสนทนาในรูปแบบเสียงได้ แต่ไม่สามารถสนทนาในรูปแบบวิดีโอได้
- สามารถ share location และ share contacts ได้
- สามารถสร้างกลุ่มเพื่อนเพื่อส่งข้อมูลหาเพื่อนเป็นกลุ่มได้



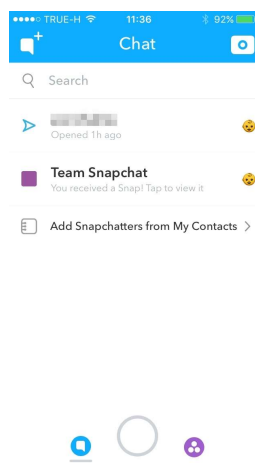
Snapchat



Snapchatเป็นapplicationที่มีความโดดเด่นด้านการส่งข้อความในรูปแบบภาพ และเมื่อถึงเวลาที่กำหนดภาพนั้นก็จะหายไปเอง ทำให้ผู้ใช้งานรู้สึกสบายใจมากขึ้นว่า ภาพที่ส่งหาเพื่อนนั้นจะไม่ถูกนำไปใช้งานหรือส่งต่อให้เจ้าของภาพเสียหาย

Snapchat บน iOS

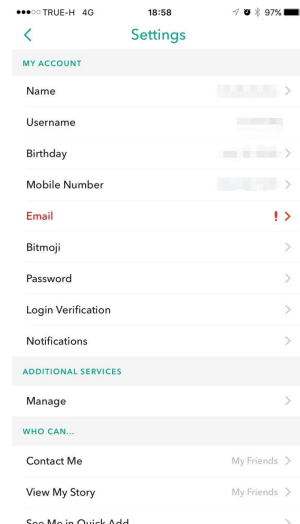
หลังจากที่เราดาวน์โหลดSnapchatจากApp Store มาติดตั้งเรียบร้อยแล้ว และเปิดเข้าใช้งานเราจะเจอกับโหมดกล้องก่อนเลยให้เราแตะที่มุมซ้ายล่างของจอเพื่อเข้าใช้งานในหน้าChat



ในหน้าChatนี้เราจะพบกับรายชื่อของเพื่อนเราที่เราได้ทำการส่งข้อมูลหาด้วย มุมซ้ายบนจะเป็นการเปิดเข้าไปในหน้าของรายชื่อของเพื่อนเรา มุมขวาบนจะเป็นการกลับไปยังหน้ากล้องถ่ายรูป ส่วนแถบเมนูด้านล่างนี้คือการเลือกว่าเราจะไปหน้าไหนของแอปพลิเคชัน ทางซ้ายคือหน้าchat ตรงกลางคือการเข้าไปที่โหมดกล้องถ่ายรูป ทางขวาจะเป็นหน้าที่แสดงตัวอย่างของการใช้งานsnapchat (หรือเราจะเรียกว่าหน้าโฆษณาก็ได้)

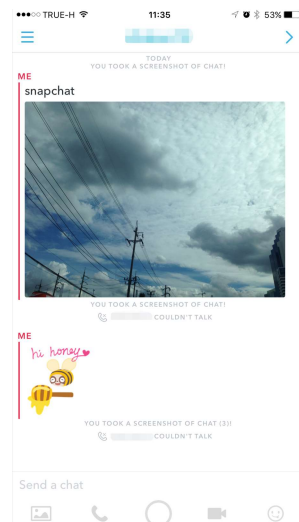
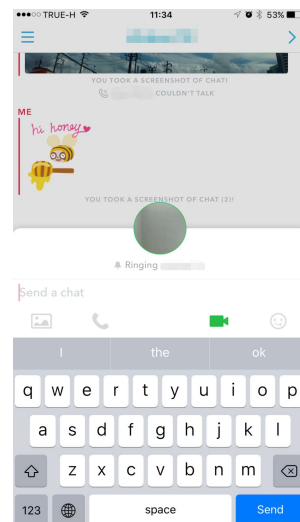
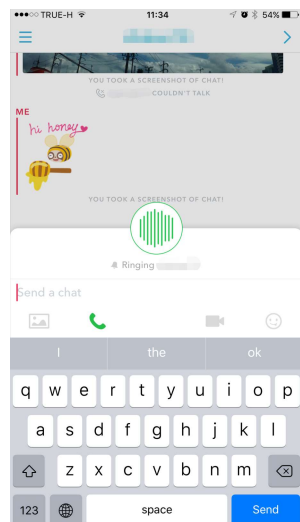
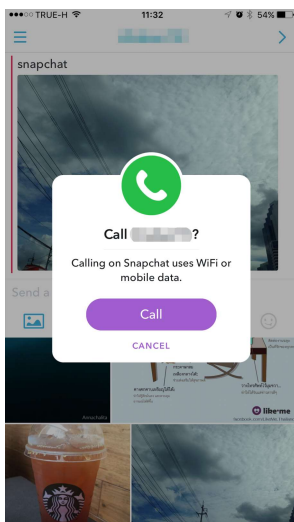
ในหน้ากล้องถ่ายรูปนั้นจะมีหน้าย่อยให้เราใช้งานได้อีก จะเป็นหน้าที่มีรายละเอียดเกี่ยวกับaccountของเรา คือ

- Added Me : จะเป็นหน้าที่บอกว่าใครขอมาเป็นเพื่อนกับเราบ้าง
- Add Friends : เราสามารถเพิ่มเพื่อนได้จากหน้านี้
- My Friends : เพื่อนของเรามีใครบ้าง
- Setting : เป็นหน้าที่บอกรายละเอียดเกี่ยวกับเรา และเราสามารถเข้าไปตั้งค่าต่างๆได้จากหน้านี้



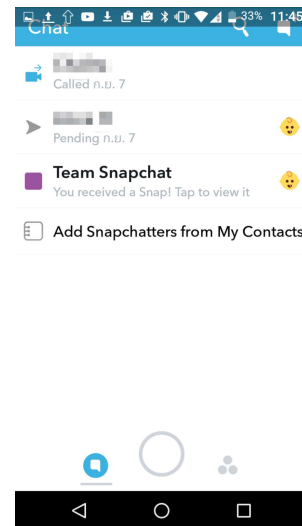
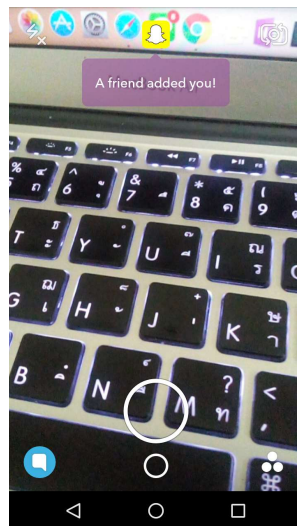
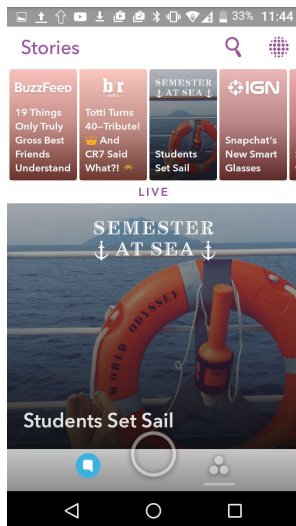
ในหน้าสนทนาในsnapchatถึงจะมีความโดดเด่นในการส่งภาพถ่าย แต่ก็มีความสามารถอื่นๆดังนี้

- สามารถส่งข้อมูลในรูปแบบอักษรได้ แต่ไม่สามารถส่งข้อมูลในรูปแบบเสียงได้
- สามารถส่งข้อมูลในรูปแบบภาพและวิดีโอได้
- สามารถส่งข้อมูลในรูปแบบสติ๊กเกอร์หรืออีโมติคอนได้
- ไม่สามารถส่งข้อมูลในรูปแบบไฟล์เอกสารได้
- สามารถสนทนาในรูปแบบเสียงและรูปแบบวิดีโอได้
- ไม่สามารถ share location และ share contacts ได้
- ไม่สามารถสร้างกลุ่มเพื่อนเพื่อส่งข้อมูลหาเพื่อนเป็นกลุ่มได้

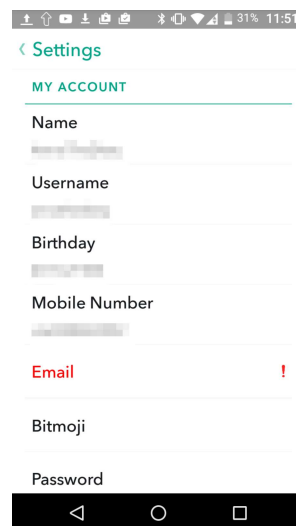
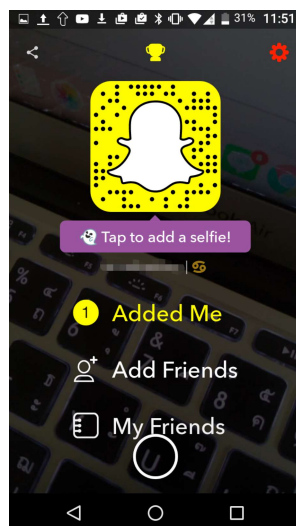


Snapchat บน Android

Snapchat เป็นแอปพลิเคชันที่ไม่มีความแตกต่างกันเลยในเวอร์ชันของ Android และ iOS โดยมีหน้าหลักๆในการใช้งาน 3 หน้า คือ หน้าChat , กล้องถ่ายรูป , หน้าตัวอย่างการใช้งาน

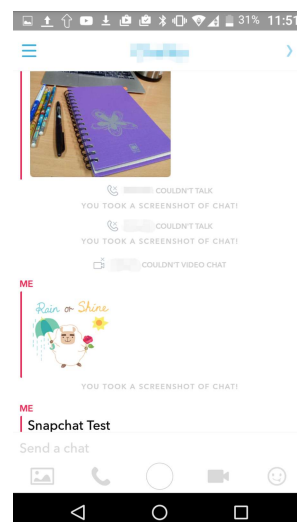
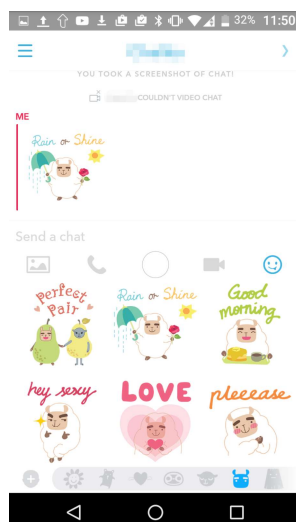
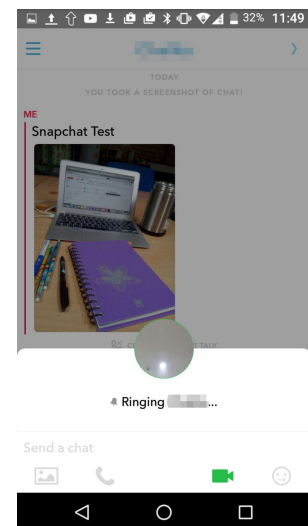
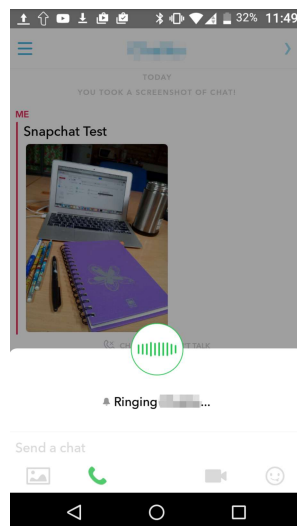
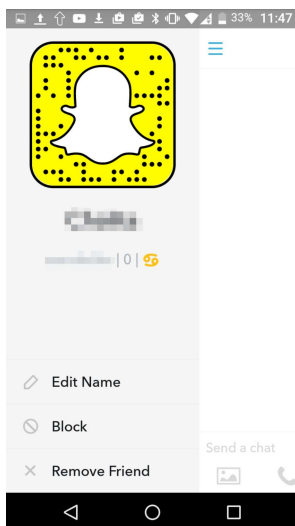


ในหน้าหลักกล้องถ่ายรูปจะมีหน้าย่อยเข้าไป จะเป็นหน้าaccount และ การตั้งค่า



ความสามารถในหน้าสนทนาของsnapchatเวอร์ชันandroidก็มีความสามารถไม่ต่างอะไรจากเวอร์ชันiosคือ

- สามารถส่งข้อมูลในรูปแบบอักษรได้ แต่ไม่สามารถส่งข้อมูลในรูปแบบเสียงได้
- สามารถส่งข้อมูลในรูปแบบภาพและวิดีโอได้
- สามารถส่งข้อมูลในรูปแบบสติ๊กเกอร์หรืออีโมติคอนได้
- ไม่สามารถส่งข้อมูลในรูปแบบไฟล์เอกสารได้
- สามารถสนทนาในรูปแบบเสียงและรูปแบบวิดีโอได้
- ไม่สามารถ share location และ share contacts ได้
- ไม่สามารถสร้างกลุ่มเพื่อนเพื่อส่งข้อมูลหาเพื่อนเป็นกลุ่มได้



ถ้าเราเปรียบเทียบ messaging app ทั้ง 6 ตัวในแง่มุมมองต่อไปนี้

- การเข้ารหัสในช่องส่งข้อความ
- ผู้ให้บริการไม่สามารถถอดรหัสเพื่อดูบทสนทนาได้
- ยืนยันตัวตนของผู้ติดต่อด้วยได้หรือไม่
- ถ้าถูกแจกอรรถหัสหาย ข้อมูลที่มีมาในอดีตจะปลอดภัย
- โปรแกรมเปิดเผยซอร์สเพื่อให้ประเมิน
- การออกแบบความปลอดภัยระบบมีความชัดเจน มีเอกสารกำกับทุกขั้นตอน
- มีการตรวจสอบซอร์สโค้ดล่าสุด

เราสามารถจะสรุปได้เป็นตารางดังต่อไปนี้

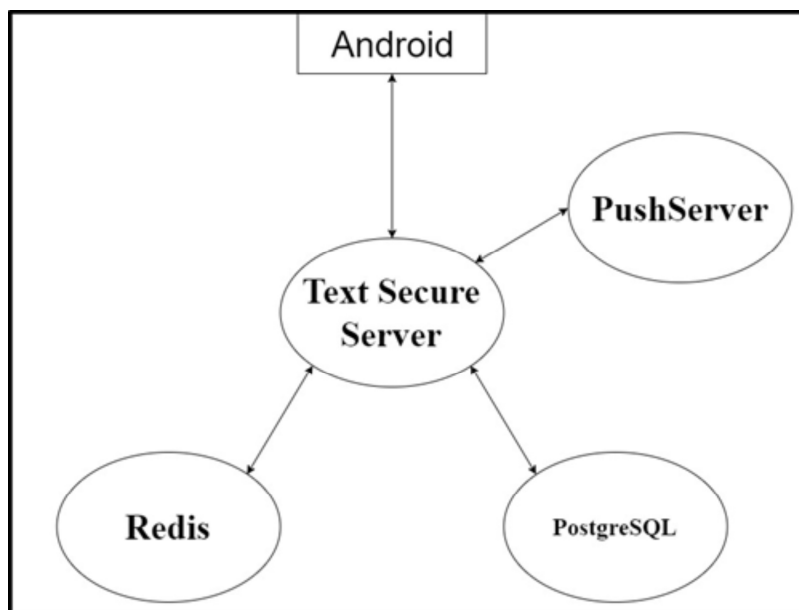
	การเข้ารหัสในช่องส่งข้อความ	ผู้ให้บริการไม่สามารถถอดรหัสเพื่อดูบทสนทนาได้	ยืนยันตัวตนของผู้ติดต่อด้วยได้	ถ้าถูกแจกอรรถหัสหาย ข้อมูลที่มีมาในอดีตจะปลอดภัย	โปรแกรมเปิดเผยซอร์สเพื่อให้ประเมิน	การออกแบบความปลอดภัยระบบมีความชัดเจน มีเอกสารกำกับทุกขั้นตอน	มีการตรวจสอบซอร์สโค้ดล่าสุด
LINE	✓	✗	✓	✗	✗	✓	✓
Skype	✓	✗	✗	✗	✗	✗	✗
WhatsApp	✓	✓	✓	✓	✗	✓	✓
Telegram	✓	✗	✗	✗	✓	✓	✓
Signal	✓	✓	✓	✓	✓	✓	✓
Snapchat	✓	✗	✗	✗	✗	✗	✓

โดยคณะผู้วิจัยเลือกจะพัฒนาต่อจาก codebase ของ Signal เพราะมีคุณสมบัติทั้ง 6 ครบถ้วน แตกต่างจาก messaging app อื่นๆที่พิจารณาที่ขาดคุณสมบัติบางตัวไป
ต่อไปจะรายงานผลการดัดแปลง ติดตั้ง และทดสอบ Signal

ผลการดัดแปลง ติดตั้ง และทดสอบ Signal

การติดตั้ง Signal Server และตั้งค่า Configuration

ส่วนประกอบโดยรวมทั้งหมดของระบบ Text Secure Server (Signal Server version 0.93) ประกอบไปด้วย Redis, PostgreSQL, PushServer และ Android (App Signal) ดังแสดงในรูปด้านล่าง



รายละเอียดการดำเนินงานดัดแปลงและติดตั้ง Signal มีดังนี้

1. ไปยัง <https://github.com/lucaconte/BeatTheMeddler> เพื่อศึกษางานที่มีผู้พัฒนาได้พัฒนาเบื้องต้นไว้บางส่วนแล้ว

2. Download หรือ Clone (หากมี Git) Signal Server ซึ่งประกอบด้วย TextSecure-Server และ PushServer จากลิงค์ต่อไปนี้

<https://github.com/WhisperSystems/TextSecure-Server> และ

<https://github.com/WhisperSystems/PushServer>

Server ที่จะทำการติดตั้ง Signal Server จะต้องมีโปรแกรมต่าง ๆ ตามด้านล่างติดตั้งไว้ด้วย

- JAVA 1.7 ที่เป็นของ OpenJdk

- Redis

- PostgreSQL (สร้าง DB accountsdb และ messaged ด้วยคำสั่ง

```
#sudo -u postgres createdb -O signal accountsdb
```

```
#sudo -u postgres createdb -O signal messagedb)
```

- Twilio account สำหรับการยืนยันตัวตนเมื่อสมัครการใช้ Signal โดยผ่าน SMS (คณะผู้จัดทำไม่ได้ติดตั้งเนื่องจากไม่ต้องการใช้บริการ)
- Amazon AWS S3 bucket service สำหรับการรับส่งไฟล์ (ผู้จัดทำไม่ได้ติดตั้งเนื่องจากไม่ต้องการใช้บริการ)
- GCM account จาก google developer console เพื่อการติดตามสำหรับแอปพลิเคชันที่นำไปใช้จริง (ผู้จัดทำไม่ได้ติดตั้งเนื่องจากไม่ต้องการใช้บริการ)
- Apple Push Notifications (APN) สำหรับ IOS (ไม่ได้ติดตั้งเนื่องจากทดสอบเฉพาะ Android)

3. ทำการเปลี่ยน APN Service ตาม Path เริ่มจาก PushServer ที่ได้ Download หรือ Clone จาก [./org/whispersystems/pushserver/senders/APNSender.class](https://github.com/whispersystems/pushserver) ใน method “public void start ()” ดังแสดงในรูปด้านล่าง เปลี่ยนจาก withProductionDestinetion() เป็น withSandboxDestination

```

//[LC] uncomment for production releases (with the right certificate)
// this.pushApnService = APNS.newService()
//
//             .withCert(new ByteArrayInputStream(pushKeyStore), "insecure")
//             .asQueued()
//             .withProductionDestination().build();

this.pushApnService = APNS.newService()
    .withCert(new ByteArrayInputStream(pushKeyStore), "insecure")
    .asQueued()
    .withSandboxDestination().build();

```

4. ทำการ compile ด้วย Command Line หลังจากแก้ไข ด้วยคำสั่ง

```

#apt-get install -y maven

#cd PushServer

#mvn clean install

```

จะพบ Error ในลักษณะตามนี้

```
[ERROR] Failed to execute goal on project TextSecureServer: Could not resolve dependencies
for project org.whispersystems.textsecure:TextSecureServer:jar:0.92:
Failed to collect dependencies at org.whispersystems:websocket-resources:jar:0.3.2: Failed to
read artifact descriptor for org.whispersystems:websocket-resources:jar:0.3.2: Could not find
artifact org.whispersystems:parent:pom:0.3.2 in gcm-server-repository (https://raw.githubusercontent.com/
whispersystems/maven/master/gcm-server/releases/) -> [Help 1]
```

ต้องทำการ Bypass Error นี้ด้วยการ Clone WebSocket-Resource จากลิงค์

<https://github.com/lucaconte/WebSocket-Resources.git>

ทำการ compile ด้วย Command Line ดังนี้

```
#cd ../WebSocket-Resources
```

```
#mvn clean install
```

```
#mvn install:install-file -Dfile=./library/target/websocket-resources-0.3.2.jar -
```

```
DgroupId=org.whispersystems -DartifactId=websocket-resources -Dversion=0.3.2 -
```

```
Dpackaging=jar
```

5. ทำการ Bypass SMS service โดยแก้ไข 3 ไฟล์

ไฟล์ที่ 1

```
src/main/java/org/whispersystems/textsecuregcm/sms/TwilioSmsSender.java ลบ
code ที่อยู่ใน Method `deliverVoxVerification`, `deliverSmsVerification` และให้
`getRandom` return ค่าคงที่ (100000)
```

ไฟล์ที่ 2

src/main/java/org/whispersystems/textsecuregcm/sms/SmsSender.java ลบ code ที่
อยู่ภายใน Method `deliverSmsVerification`

ไฟล์ที่ 3

src/main/java/org/whispersystems/textsecuregcm/controllers/
AccountController.java ใน Method createAccount() ภายใน if else ให้ลบ code ที่อยู่ภายใน
ออก จะได้ดังนี้

```
if (testDevices.containsKey(number)) {
    // noop
} else if (transport.equals("sms")) {
} else if (transport.equals("voice")) {}
```

ใน Method generateVerificationCode (Constant) แก้ไขค่า randomInt เป็น 100000
ได้ดังนี้ int randomInt = 100000;

6. จากนั้น compile TextSecureServer ด้วย คำสั่งดังนี้

```
#cd ..
#cd TextSecure-Server
#mvn install
```

หากพบปัญหาที่เกี่ยวกับการทดสอบ จะทำการข้ามการทดสอบไปด้วยการเพิ่ม

-DskipTests จาก mvn install

7. แก้ Config ใน File config/textsisure.yml ดังนี้

```
twilio:
  accountId: AC302d9ea2695e21cd17ce15bc510d28fd #fake
  accountToken: febf5ccba3b4051dd7e7d0901a0fd404 #fake
  numbers:
    -
      +66876157370 #fake
  localDomain: domain
push:
  host: localhost
  port: 9090
  username: whisper
  password: thepassword
s3:
  accessKey: ABCDEFGCUFYDHVM2LXXX #fake
  accessSecret: W0UfGDddfAbqYyCTIiBSQlDtreTGokOs00TpL0SE #fake
  attachmentsBucket: thenameofyouts3buket #fake
directory:
  url: "redis://localhost:6379/0"
cache:
  url: "redis://localhost:6379/1"
server:
  applicationConnectors:
    - type: https
      port: 8080
      keyStorePath: ./ssl/example.keystore
```

```
keyStorePassword: example
#keyStoreType : PKCS12
  validateCerts: false
adminConnectors:
- type: https
  port: 8081
  keyStorePath: ./ssl/example.keystore
  keyStorePassword: example
  #keyStoreType : PKCS12
  validateCerts: false
websocket:
  enabled: true
messageStore: # Postgres database configuration for message store
  driverClass: org.postgresql.Driver
  user: "whisper"
  password: "thepassword"
  url: "jdbc:postgresql://localhost:5432/messagedb"
database:
  driverClass: org.postgresql.Driver
  user: "whisper"
  password: "thepassword"
  url: "jdbc:postgresql://localhost:5432/accountdb"
  properties:
    charSet: UTF-8
#federation: # is disabled
```

```

logging:
  level: INFO
appenders:
  - type: file
    currentLogFilename: /tmp/textsecurshserver.log
    archivedLogFilenamePattern: /temp/textsecurserver-%d.log.gz
    archivedFileCount: 5
  - type: console
redphone:
  authKey: 1234567890 #fake

```

8. แก้ Config ใน File config/pushserver.yml ดังนี้

```

redis:
  url: redis://localhost:6379/2
authentication:
  servers:
    -
      name: whisper
      password: thepassword
gcm:
  xmpp: false
  apiKey: AIzaSyC8gPzceq2SPebZZWaD3I9OeqePyD9CUqk #real
  senderId: 412918270132 #real
  redphoneApiKey: AIddSyAsviyMy8jKe8chCEfr8NbeqGghy7oOCi4 #fake

```

```
server:
  applicationConnectors:
    - type: http
      port: 9090
  #keyStorePath: ./ssl/example.keystore
  #keyStorePassword: example
  #keyStoreType : PKCS12
  #validateCerts: false
  adminConnectors:
    - type: http
      port: 9091
      #keyStorePath: ./ssl/example.keystore
      #keyStorePassword: example
      #keyStoreType : PKCS12
      #validateCerts: false
  gzip:
    enabled: true
logging:
  level: INFO
  appenders:
    - type: file
      currentLogFilename: /tmp/pushserver.log
      archivedLogFilenamePattern: /tmp/pushserver-%d.log.gz
      archivedFileCount: 5
    - type: console
```


9. จากนั้นติดตั้ง PostgreSQL Database โดยใช้คำสั่งต่อไปนี้

```
#apt-get update

#apt-get install -y redis-server postgresql openjdk-7jre-headless

#/etc/init.d/postgresql start #sudo -u postgres psql --command "CREATE USER
signal WITH SUPERUSER PASSWORD 'thepassword';"

#sudo -u postgres createdb -O signal accountsdb

#sudo -u postgres createdb -O signal messaged
```

10. Migrate config ของ TextSecure เข้ากับ Database ตามคำสั่งดังนี้

```
#java -jar TextSecure-Server/target/ TextSecureServer-0.93_sin.jar accountsdb
migrate config/textsecure.yml

#java -jar TextSecure-Server/target/ TextSecureServer-0.93_sin.jar messagedb
migrate
```

11. Remote Login (ผ่าน Secure Shell) ไปยัง Server เพื่อแก้ไขให้ Signal รองรับ HTTPS ได้ ด้วยคำสั่งต่อไปนี้ (Code ด้านล่างเป็น Bash Command สามารถ copy ไปใส่ไฟล์ .sh แล้วสั่ง Run Script ได้เลย)

```
#!/bin/bash
#This script creates root CA and server certificates to be used by the
client and the server.
# rootCA.crt needs to be copied to the client to replace the system-
wide root CA set
# example.keystore needs to be referenced by keyStorePath in the
server's config file
#
#TO EXECUTE FROM THE OUTSIDE WITH:
#ALTNAME=DNS:signal.foo.org ./gencerts
#
#IF YOU HAVE A DOMAIN OR IP if U are in a lan
#ALTNAME=IP:<IP ของเครื่อง Server > ./gencerts
#
```

```
# Create private key for root CA certificate
openssl genrsa -out rootCA.key 4096

# Create a self-signed root CA certificate
openssl req -x509 -new -nodes -days 3650 -out rootCA.crt -key
rootCA.key

# Create server certificate key
openssl genrsa -out whisper.key 4096

# Create Certificate Signing Request
openssl req -new -key whisper.key -out whisper.csr

# Sign the certificate with the root CA
openssl x509 -req -in whisper.csr -CA rootCA.crt -CAkey rootCA.key -
CAcreateserial -days 365 -out whisper.crt -extensions extensions -
extfile <(cat <<-EOF
[ extensions ]
basicConstraints=CA:FALSE
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer
subjectAltName=IP:<IP ของเครื่อง Server > ./gencerts
EOF)
```

```
# Export to host key and certificate to PKCS12 format which is
recognized by Java keytool
openssl pkcs12 -export -password pass:example -in whisper.crt -inkey
whisper.key -out keystore.p12 -name example -CAfile rootCA.crt

# Import the host key and certificate to Java keystore format, so it
can be used by dropwizard
keytool -importkeystore -srcstoretype PKCS12 -srckeystore keystore.p12
-srcstorepass example -destkeystore example.keystore -deststorepass
example

#whisper.store must be placed into Android client
# -- keytool -importcert -v -trustcacerts -file whisper.crt -alias
IntermediateCA -keystore whisper.store -provider
org.bouncycastle.jce.provider.BouncyCastleProvider -providerpath ../
bcprov-jdk15on-154.jar -storetype BKS -storepass whisper

#for iOS client you have to convert whisper.crt in DER format and
install whisper.cer in Signal-iOS
#openssl x509 -in whisper.crt -out whisper.cer -outform DER
```

12. Run Server ตามคำสั่งต่อไปนี้

```
#!/etc/init.d/postgresql start
```

```
#rm -rf /var/run/redis_6379.pid
```

```
#service redis_6379 start
```

```
#java -jar PushServer/target/Push-Server-0.12.0capsule-fat.jar server  
pushserver.yml &
```

```
#java -jar TextSecure-Server/target/ TextSecureServer-0.93_sin.jar server  
textsecure.yml
```

จะสังเกตว่าเซอรัวิสต่างๆที่เกี่ยวข้องเริ่มทำงานขึ้นมาตามชุดภาพด้านล่าง

```
root@18e4347139c0:~# /etc/init.d/postgresql start  
* Starting PostgreSQL 9.3 database server  
root@18e4347139c0:~# █ [ OK ]
```

Postgresql Start

```
root@18e4347139c0:~# rm -rf /var/run/redis_6379.pid  
root@18e4347139c0:~# service redis_6379 start  
Starting Redis server...  
root@18e4347139c0:~# █
```

Redis Start

หลังจาก Server Start เสร็จแล้ว ทดสอบอย่างง่ายด้วยการเข้าผ่าน Browser <https://ip:8081> เช่น <https://192.168.1.8:8081>



หน้าเว็บที่แสดงว่า Server ใช้งานได้

```

"GET / HTTP/1.1" 404 284 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.81 Safari/537.36" 55
"GET /favicon.ico HTTP/1.1" 404 295 "https://192.168.1.8:8080/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/5
"GET / HTTP/1.1" 200 - "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.81 Safari/537.36" 2
"GET /favicon.ico HTTP/1.1" 404 295 "https://192.168.1.8:8081/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/5
"GET /api/v1/feedback/gcm HTTP/1.1" 200 14 "-" "whisper-server (whisper-server)" 4
"GET /api/v1/feedback/apn HTTP/1.1" 200 14 "-" "whisper-server (whisper-server)" 2
"GET /metrics?pretty=true HTTP/1.1" 200 - "https://192.168.1.8:8081/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_4) AppleWebKit/537.36 (KHTML, like Gecko) Ch
  
```

Log จาก Server ว่ามีการเชื่อมต่อ

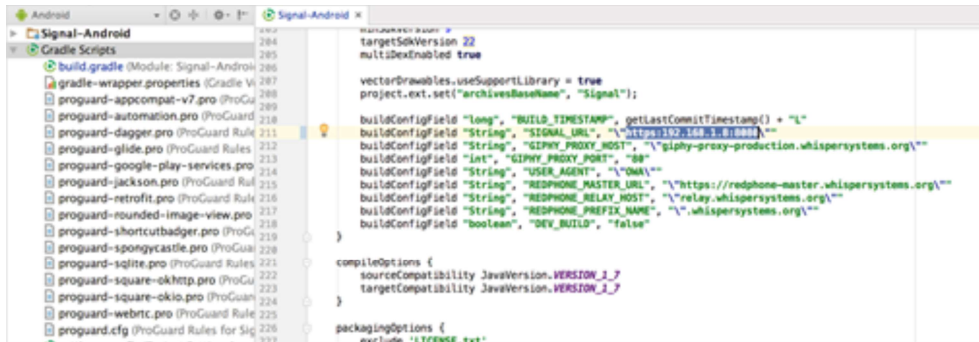
เมื่อทำงานได้ตามข้างต้น แสดงว่า Server พร้อมใช้งานแล้ว

การตั้งค่าแอปพลิเคชัน Signal Android เพื่อเชื่อมต่อกับ Signal Server

1. ดาวน์โหลด Source file ของ Signal Android จาก ลิงค์

<https://github.com/WhisperSystems/Signal-Android>

2. เปิด Source ที่ Download หรือ Clone มาด้วย Android Studio
3. ทำการแก้ไข buildConfigField "String", "SIGNAL_URL", "\"https://textsecure-service.whispersystems.org\"" เป็น IP ของ Server ที่เราตั้งขึ้นตามรูปด้านล่าง



```
284 targetSdkVersion 22
285 multiDexEnabled true
286
287 vectorDrawables.useSupportLibrary = true
288 project.ext.set("archivesBaseName", "Signal");
289
290 buildConfigField "long", "BUILD_TIMESTAMP", getLatestCommitTimestamp() + "L"
291 buildConfigField "String", "SIGNAL_URL", "\"https://textsecure-service.whispersystems.org\""
292 buildConfigField "String", "SIPHY_PROXY_HOST", "\"siphy-proxy-production.whispersystems.org\""
293 buildConfigField "int", "SIPHY_PROXY_PORT", "88"
294 buildConfigField "String", "USER_AGENT", "\"OMA\""
295 buildConfigField "String", "REDPHONE_MASTER_URL", "\"https://redphone-master.whispersystems.org\""
296 buildConfigField "String", "REDPHONE_RELAY_HOST", "\"relay.whispersystems.org\""
297 buildConfigField "String", "REDPHONE_PREFIX_NAME", "\".whispersystems.org\""
298 buildConfigField "boolean", "DEV_BUILD", "false"
299
300 }
301
302 compileOptions {
303     sourceCompatibility JavaVersion.VERSION_1_7
304     targetCompatibility JavaVersion.VERSION_1_7
305 }
306
307 packagingOptions {
308     exclude "LICENSE.txt"
309 }
```

4. ไปยังไฟล์ RegistrationService.java ใน Folder

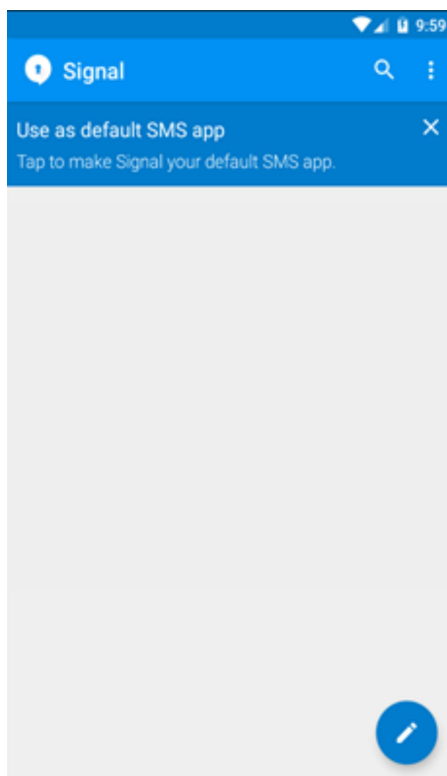
src>org>thoughtcrime>securesms>service>RegistrationService ปิด GCM โดยการ comment ดังแสดง
ในรูปด้านล่าง

```

261     } else {
262         TextSecurePreferences.setGcmDisabled(this, true);
263     }
264     TextSecurePreferences.setWebsocketRegistered(this, true);
265     DatabaseFactory.getIdentityDatabase(this).saveIdentity(self.getRecipientId(), identityKey.getPublicKey());
266     DirectoryHelper.refreshDirectory(this, accountManager, number);
267     /*if (supportsGcm) {
268         RedPhoneAccountManager redPhoneAccountManager = new RedPhoneAccountManager(BuildConfig.REDPHONE_MASTER_URL,
269             new RedPhoneTrustStore(this),
270             number, password);
271         String verificationToken = accountManager.getAccountVerificationToken();
272         redPhoneAccountManager.createAccount(verificationToken, new RedPhoneAccountAttributes(signalKey, TextSecurePreferences.getGcmRegistrationKey()));
273     }*/
274     DirectoryRefreshListener.schedule(this);
275     RotateSignedPreKeyListener.schedule(this);
276     private synchronized String waitForChallenge() throws AccountVerificationTimeoutException {
277         this.verificationStartTime = System.currentTimeMillis();
278         if (this.challenge == null) {
279             try {

```

5. Run Source ใน Emu หรือ สร้างเป็นไฟล์ .APK เพื่อติดตั้งบนสมาร์ตโฟนที่มีระบบปฏิบัติการแอนดรอยด์ 2 เครื่อง
6. จากนั้นกรอกข้อมูลหมายเลขโทรศัพท์ลงไป เพื่อลงทะเบียนกับ Server
7. ระบบจะทำการส่ง SMS มายังหมายเลขที่กรอกไว้ แต่ในกรณีทดสอบ เราได้ทำการ Bypass การส่ง SMS เนื่องจากมีค่าใช้จ่าย จึงรอให้หมดเวลา
8. หลังจากหมดเวลา ระบบจะเปลี่ยนไปอีกหน้า เราสามารถแก้ไขหมายเลขเพื่อไปขั้นตอนที่ 6 อีกครั้ง หรือจริง ๆ แล้ว Signal ยอมให้ผู้ใช้กด “CALL ME” เพื่อฟังรหัสผ่านได้ โดย Signal จะโทรมาที่โทรศัพท์ตามเบอร์โทรศัพท์ที่ให้ไว้ในขั้นตอนที่ 7 แต่ในโครงการนี้ ผู้จัดทำโครงการได้ละขั้นตอนนี้ไว้ จากนั้นผู้ใช้กรอกรหัสที่ตั้งไว้ตอนแก้ไขไฟล์ AccountController.java ใน Server คือ 100000
9. จากนั้นทำการรอ และเข้าสู่หน้าที่สามารถใช้งานได้ จะเห็นหน้าจอพร้อมใช้งานตามรูปด้านล่าง

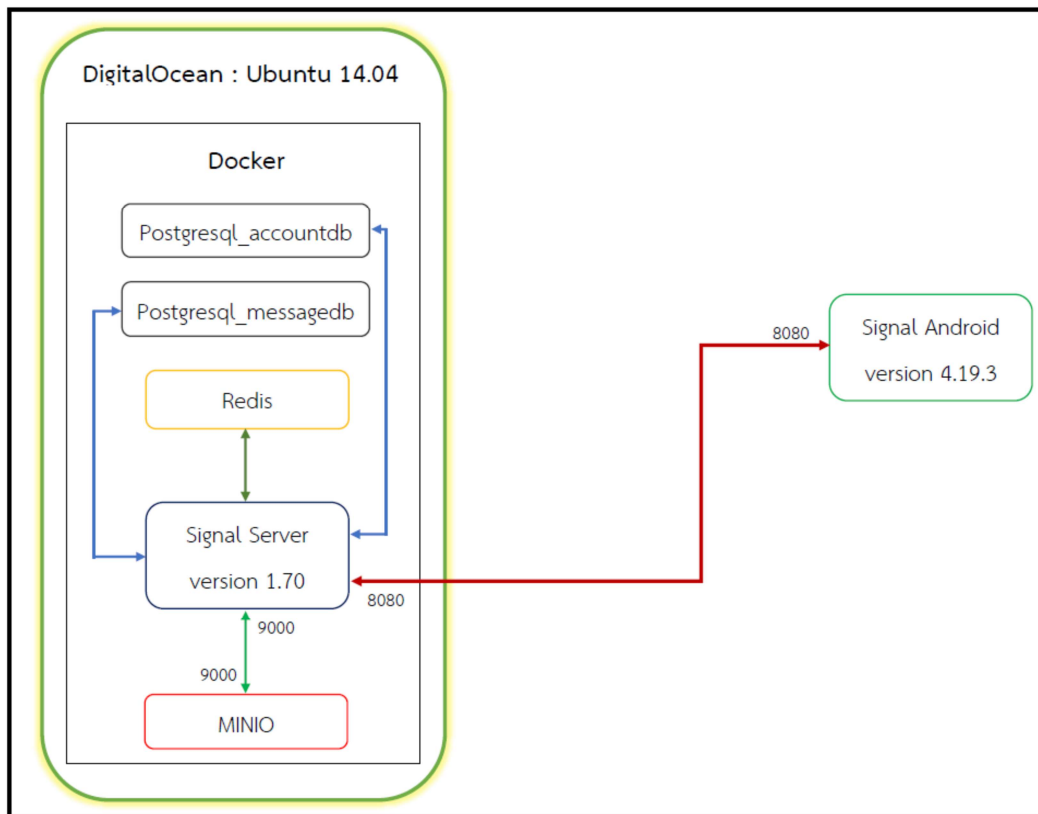


10. ทดสอบการใช้งานโดยการส่งข้อความไปที่สมาร์ทโฟนเครื่องที่สอง (ให้ค้นหาคู่สนทนาด้วยหมายเลขโทรศัพท์) แอปพลิเคชันบนแอนดรอยด์พร้อมที่จะใช้ในการทดสอบแล้ว

การเตรียมเครื่อง Server บน DigitalOcean

เพื่อให้ผู้ใช้ App ส่งข้อมูลผ่านเครือข่ายได้ ซอฟต์แวร์แม่ข่ายจำเป็นต้องมีที่อยู่ไอพีสาธารณะ (Public IP Address) ดังนั้น คณะผู้จัดทำจึงเลือกให้มี ซอฟต์แวร์แม่ข่ายสำหรับการทำงานของ Docker Container ที่อยู่ใน DigitalOcean ซึ่งหากผู้ดูแลต้องการย้าย Container ให้ไปอยู่ในเครื่องแม่ข่ายภายในองค์กรของตนเองภายหลัง ก็ทำได้ง่ายและรวดเร็ว

ส่วนประกอบโดยรวมทั้งหมดของระบบ Signal Server บน DigitalOcean ประกอบไปด้วย Redis, PostgreSQL_accountdb, PostgreSQL_messagedb, Signal Server, MINIO และ Signal Android ตามรูปด้านล่าง



1. ติดตั้ง Docker และ Git บน DigitalOcean ขั้นตอนนี้ผู้ใช้จะต้องมีบัญชี Git ที่ Github หรือผู้ให้บริการ Git online อื่นๆ และมี directory ที่เก็บ code ของ Signal server ไว้แล้ว
2. สร้าง directory ชื่อ docker-Signal-Server และสร้างไฟล์ docker-compose.yml ขึ้นมา ซึ่งภายในไฟล์มีลักษณะตามภาพด้านล่าง

```
1 |version: '2'
2 |services:
3 |
4 |  postgresql_accountdb:
5 |    image: postgres:9.3.20
6 |    container_name: docker_postgresql_accountdb
7 |    restart: always
8 |    volumes:
9 |      - ./postgresql/accountdb.sql:/docker-entrypoint-initdb.d/accountdb.sql
10 |     - ./postgresql/accountdb:/var/lib/postgresql/data/
11 |    # ports:
12 |    #   - "5421:5432"
13 |    environment:
14 |      - POSTGRES_USER=signal
15 |      - POSTGRES_PASSWORD=thepassword
16 |      - POSTGRES_DB=accountdb
17 |
18 |  postgresql_messedb:
19 |    image: postgres:9.3.20
20 |    container_name: docker_postgresql_messedb
21 |    restart: always
22 |    volumes:
23 |      - ./postgresql/messedb.sql:/docker-entrypoint-initdb.d/messedb.sql
24 |     - ./postgresql/messedb:/var/lib/postgresql/data/
25 |    # ports:
26 |    #   - "5422:5432"
27 |    environment:
28 |      - POSTGRES_USER=signal
29 |      - POSTGRES_PASSWORD=thepassword
30 |      - POSTGRES_DB=messedb
31 |
32 |  reedit:
33 |    image: redis:3.2.8
34 |    container_name: docker_redis
35 |    restart: always
36 |    ports:
37 |      - "6379:6379"
```

```
39 signalserver:
40   build: ./signalserver
41   container_name: docker_signal-server
42   restart: always
43   ports:
44     - "8080:8080"
45     - "8081:8081"
46
47 minio:
48   image: minio/minio
49   ports:
50     - "9000:9000"
51   volumes:
52     - ./minio/.minio/data:/export
53     - ./minio/.minio/config:/root/.minio
54   environment:
55     - "MINIO_ACCESS_KEY=AKIAIG4ILCORMAJCS37A"
56     - "MINIO_SECRET_KEY=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"
57   command: server /export
58
59 createbuckets:
60   image: minio/mc
61   depends_on:
62     - minio
63   entrypoint: >
64     /bin/sh -c "
65     /usr/bin/mc config host add myminio https://minio:9000 AKIAIG4ILCORMAJCS37A
66     u8cQx07PvHJS8/zvr7q3IFY+w2toIYIJQ7vm1ETH;
67     /usr/bin/mc rm -r --force myminio/signal-attachments-buu;
68     /usr/bin/mc rm -r --force myminio/signal-profiles-buu;
69     /usr/bin/mc mb myminio/signal-attachments-buu;
70     /usr/bin/mc mb myminio/signal-profiles-buu;
71     /usr/bin/mc policy public myminio/signal-attachments-buu;
72     /usr/bin/mc policy public myminio/signal-profiles-buu;
73     exit 0;
74     "
```

3. สร้าง directory ชื่อ signalserver ไว้ใน directory docker-Signal-Server
4. เข้าไป Signal-Server ที่ได้ และทำการแก้ไขไฟล์สำหรับ ปิดการใช้งาน apn

ไฟล์ที่ 1

แก้ไขไฟล์ WhisperServerConfiguration.java

ที่ /Signal-Server/src/main/java/org/whispersystems/textsecuregcm

ลบ Code @NotNull ที่ตัวแปร private ApnConfiguration apn;

ไฟล์ที่ 2

แก้ไขไฟล์ WhisperServerService.java

ที่ /Signal-Server/src/main/java/org/whispersystems/textsecuregcm

ลบ Code ดังนี้

```
- APNSender apnSender = new
```

```
    APNSender(accountsManager,config.getApnConfiguration());
```

```
- ApnFallbackManager apnFallbackManager = new
```

```
    ApnFallbackManager(apnSender, pubSubManager);
```

```
- apnSender.setApnFallbackManager(apnFallbackManager);
```

```
- environment.lifecycle().manage(apnFallbackManager);
```

ใส่ null แทน apnFallbackManager และ apnSender ที่

```
- PushSender pushSender = new PushSender(apnFallbackManager,
```

```
    gcmSender, apnSender, websocketSender,
```

```
    config.getPushConfiguration().getQueueSize());
```

จะได้ดังนี้

```
- PushSender pushSender = new PushSender(null, gcmSender, null,
```

```
    websocketSender, config.getPushConfiguration().getQueueSize());
```

ไฟล์ที่ 3

แก้ไขไฟล์ PushSender.java

ที่ /Signal-Server/src/main/java/org/whispersystems/textsecuregcm
/push

ลบ Code ดังนี้

- ใน Method start() ลบ apnSender.start();

- ใน Method stop() ลบ apnSender.stop();

5. เข้าไป Signal-Server ที่ได้ และทำการแก้ไขไฟล์สำหรับทำ Bypass SMS service

ไฟล์ที่ 1

แก้ไขไฟล์ AccountController.java

ที่ /Signal-Server/src/main/java/org/whispersystems/textsecuregcm/
controllers

ลบ Code ใน Method createAccount ภายใน if-else และใน Method
generateVerificationCode() ให้ int randomInt = 100000;

ไฟล์ที่ 2

แก้ไขไฟล์ TwilioSmsSender.java

ที่ /Signal-Server/src/main/java/org/whispersystems/textsecuregcm/sms

ลบ Code ใน Method deliverVoxVerification และ deliverSmsVerification

ใน Method getRandom() ให้ return elements.get(100000);

ไฟล์ที่ 3

แก้ไขไฟล์ SmsSender.java

ที่ /Signal-Server/src/main/java/org/whispersystems/textsecuregcm/sms

ลบ Code ใน Method deliverSmsVerification

6. เข้าไป Signal-Server ที่ได้ และแก้ไขไฟล์สำหรับการย้ายพื้นที่จัดเก็บข้อมูลจาก Amazon S3 ไปยัง Minio

แก้ไขไฟล์ UrlSigner.java

ที่ /Signal-Server/src/main/java/org/whispersystems/textsecuregcm/s3

เพิ่ม Code ที่ส่วนด้านบนของไฟล์ ดังนี้

- import com.amazonaws.AmazonClientException;
- import com.amazonaws.AmazonServiceException;
- import com.amazonaws.auth.AWSStaticCredentialsProvider;
- import com.amazonaws.client.builder.AwsClientBuilder;
- import com.amazonaws.ClientConfiguration;
- import com.amazonaws.regions.Regions;
- import com.amazonaws.regions.Region;
- import com.amazonaws.services.s3.AmazonS3ClientBuilder;

Code ใน Method getPreSignedUrl แก้ไขให้เป็นอย่างนี้

```
ClientConfiguration clientConfiguration = new ClientConfiguration();
clientConfiguration.setSignerOverride("AWSS3V4SignerType");
AmazonS3 client = new AmazonS3Client(credentials, clientConfiguration);
Region usEast1 = Region.getRegion(Regions.US_EAST_1);
client.setRegion(usEast1);
client.setEndpoint("{ใส่ URL ของ MINIO}");
final S3ClientOptions clientOptions = S3ClientOptions.builder().setPathStyleAccess(true).build();
client.setS3ClientOptions(clientOptions);
GeneratePresignedUrlRequest request = new GeneratePresignedUrlRequest(bucket,
String.valueOf(attachmentId), method);
```

```
request.setExpiration(new Date(System.currentTimeMillis() + DURATION));
request.setContentType("application/octet-stream");
return client.generatePresignedUrl(request);
```

7. ไปที่ directory Signal-Server/config ภายในไฟล์ Signal.yml ใส่ค่าดังภาพด้านล่างต่อไปนี้

```
1 twilio: # Twilio gateway configuration
2   accountId: AC302d9ea2695e21cd17ce15bc510d28fd #fake
3   accountToken: febf5ccba3b4051dd7e7d0901a0fd404 #fake
4   numbers: # Numbers allocated in Twilio
5     - # First number
6       +66876157370 #fake
7   # messagingServicesId:
8   localDomain: domain # Domain Twilio can connect back to for
9
10  push:
11   queueSize: 200 # Size of push pending queue
12
13  # redphone:
14  #   authKey: 1234567890 # Deprecated
15
16  server:
17   applicationConnectors:
18     - type: https
19       port: 8080
20       keyStorePath: ./ssl/example.keystore
21       keyStorePassword: example
22       #keyStoreType : PKCS12
23       validateCerts: false
24   adminConnectors:
25     - type: https
26       port: 8081
27       keyStorePath: ./ssl/example.keystore
28       keyStorePassword: example
29       #keyStoreType : PKCS12
30       validateCerts: false
31
32  turn: # TURN server configuration
33   secret: 121654fjdfgdyesdfgh # TURN server secret
34   uris:
35     - stun:yourdomain:80 #fake
36     - stun:yourdomain.com:443 #fake
37     - turn:yourdomain:443?transport=udp #fake
38     - turn:etc.com:80?transport=udp #fake
39
```



```
40 cache: # Redis server configuration for cache cluster
41 | url: "redis://redit:6379/1"
42
43 directory: # Redis server configuration for directory cluster
44 | url: "redis://redit:6379/0"
45
46 messageStore: # Postgresql database configuration for message store
47 | driverClass: org.postgresql.Driver
48 | user: "signal"
49 | password: "thepassword"
50 | url: "jdbc:postgresql://postgresql_messedb/messedb"
51
52 attachments: # MINIO configuration
53 | accessKey: AKIAIG4ILCORMAJCS37A
54 | accessSecret: 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
55 | bucket: signal-attachments-buu
56
57 profiles: # MINIO configuration
58 | accessKey: AKIAIG4ILCORMAJCS37A
59 | accessSecret: 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
60 | bucket: signal-profiles-buu
61 | region: us-east-1
62
63 database: # Postgresql database configuration
64 | driverClass: org.postgresql.Driver
65 | user: "signal"
66 | password: "thepassword"
67 | url: "jdbc:postgresql://postgresql_accountdb/accountdb"
68 | properties:
69 | | charSet: UTF-8
70
71 # #apn: # Apple Push Notifications configuration
72 # | bundleId:
73 # | pushCertificate:
74 # | pushKey:
75
76 gcm: # GCM Configuration
77 | senderId: 412918270132 #fake
78 | apiKey: AIzaSyC8gPzceq2SPebZZWaD3I9OeqePyD9CUqk #fake
79
80 logging:
81 | level: INFO
82 | appenders:
83 | | - type: file
84 | | | currentLogFilename: /tmp/textseureshserver.log
85 | | | archivedLogFilenamePattern: /tmp/textseureserver-%d.log.gz
86 | | | archivedFileCount: 5
87 | | - type: console
88
```

8. สร้างไฟล์ dockerfile ไว้ใน directory signalserver ซึ่งภายในไฟล์ลักษณะดังภาพต่อไปนี้

```
FROM ubuntu:14.04
LABEL maintainer="57160608@go.buu.ac.th"

RUN rm /bin/sh && ln -s /bin/bash /bin/sh
RUN apt-get update
RUN apt-get install wget -y

# --> Install java 1.8 <--
RUN apt-get install software-properties-common -y
RUN add-apt-repository ppa:openjdk-r/ppa
RUN apt-get -y upgrade
RUN apt-get -y update
RUN apt-get install openjdk-8-jdk openjdk-8-jre -y
RUN echo "JAVA_HOME=\"/usr/lib/jvm/java-8-openjdk-amd64\" >> /etc/environment
RUN ["/bin/bash", "-c", "source /etc/environment"]
RUN echo "export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64" >> ~/.bashrc
RUN echo "export PATH=\$JAVA_HOME/bin:\$PATH" >> ~/.bashrc
RUN ["/bin/bash", "-c", "source ~/.bashrc"]
RUN update-ca-certificates -f
RUN apt-get install --reinstall ca-certificates-java

# --> Install maven 3.5 <--
RUN apt-get install maven -y
RUN cd /usr/local && wget http://www-eu.apache.org/dist/maven/maven-3/3.5.2/binaries/apache-maven-3.5.2-bin.tar.gz
RUN cd /usr/local && tar xzf apache-maven-3.5.2-bin.tar.gz \
  && ln -s apache-maven-3.5.2 apache-maven
RUN echo "export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64" >> /etc/profile.d/apache-maven.sh
RUN echo "export M2_HOME=/usr/local/apache-maven" >> /etc/profile.d/apache-maven.sh
RUN echo "export MAVEN_HOME=/usr/local/apache-maven" >> /etc/profile.d/apache-maven.sh
RUN echo "export PATH=\${M2_HOME}/bin:\${PATH}" >> /etc/profile.d/apache-maven.sh
RUN source /etc/profile.d/apache-maven.sh && mvn -version
RUN java -version

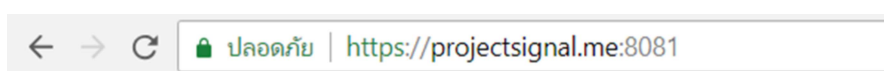
COPY ./ssl /ssl

# --> Install Signal Server <--
COPY ./Signal-Server /Signal-Server
RUN source /etc/profile.d/apache-maven.sh && cd /Signal-Server && mvn install -DskipTests

# --> Run server <--
EXPOSE 8080
EXPOSE 8081

CMD ["/bin/bash", "-c", "java -jar /Signal-Server/target/TextSecureServer-1.70.jar server /Signal-Server/config/Signal.yml"]
```

9. ใช้ certbot ในการสร้าง certificate ssl (จะต้องมี domain จริง) จะได้ไฟล์ fullchain.pem และ privkey.pem
10. สร้าง ssl directory ไว้ใน directory signalserver
 - คัดลอกไฟล์ fullchain.pem มาไว้ใน directory ssl และเปลี่ยนชื่อเป็น whisper.crt
 - คัดลอกไฟล์ privkey.pem มาไว้ใน directory ssl และเปลี่ยนชื่อเป็น whisper.key
 - ทำการรัน sslscript.sh โดยใช้คำสั่ง bash sslscript.sh
11. สร้าง directory Minio/.Minio/config/certs
 - คัดลอกไฟล์ fullchain.pem มาไว้ใน Minio/.Minio/config/certs และเปลี่ยนชื่อเป็น public.crt
 - คัดลอกไฟล์ privkey.pem มาไว้ใน Minio/.Minio/config/certs และเปลี่ยนชื่อเป็น private.key
12. ทำการรัน docker-compose up --build -d เพื่อเริ่มต้นการทำงาน ถ้า server ทำงานปกติ ไม่มีข้อผิดพลาดจะได้ภาพด้านล่างต่อไปนี้
โดยตรวจสอบจาก `https://{domainname}:8081`

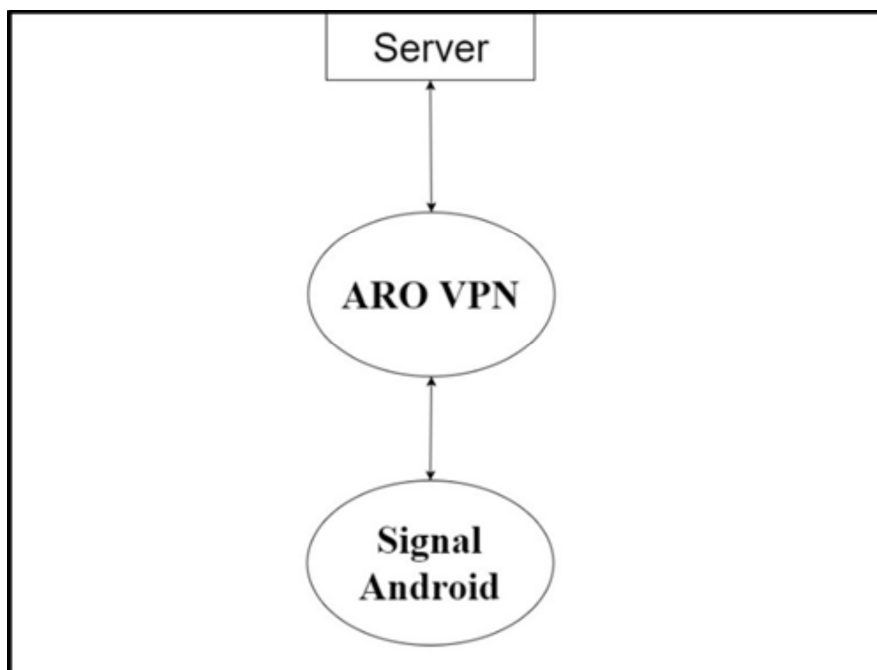


Operational Menu













- [Metrics](#)
- [Ping](#)
- [Threads](#)
- [Healthcheck](#)
- [CPU Profile](#)
- [CPU Contention](#)









การทดสอบผ่าน AT&T Application Resource Optimizer (ARO)

ARO VPN จะดักจับข้อมูลระหว่าง Signal Android กับ Server เพื่อใช้ในการทดสอบโดย ARO ดังแสดง
ในรูปด้านล่าง



ผลการทดสอบจาก ARO เป็นการบันทึกวิเคราะห์ และสรุปผลการทดสอบจากการทดสอบจำนวน 3 ครั้ง ต่อเนื่องกัน โดยในส่วนของอายุ Cache จะเริ่มนับอายุจากการทดลองครั้งที่ 3 เสร็จ หากมีครั้งใดที่ไม่ผ่าน (1 ใน 3 ครั้ง) การทดสอบนั้นจะถือว่าไม่ผ่านการทดสอบ และผลการทดสอบจาก ARO เป็นดังนี้

TESTS CONDUCTED	
	File Download: Text File Compression
	File Download: Duplicate Content
	File Download: Cache Control
	File Download: Content Expiration
	File Download: Combine JS and CSS Requests
	File Download: Resize Images for Mobile
	File Download: Image Metadata
	File Download: Image Compression
	File Download: Minify CSS, JS, JSON and HTML
	File Download: Use CSS Sprites for Images
	Connections: Connection Opening
	Connections: Unnecessary Connections - Multiple Simultaneous Connections

	Connections: Inefficient Connections - Periodic Transfers
	Connections: Inefficient Connections - Screen Rotation
	Connections: Inefficient Connections - Connection Closing Problems
	Connections: 400, 500 HTTP Status Response Codes
	Connections: 301, 302 HTTP Status Response Codes
	Connections: 3rd Party Scripts
	HTML: Asynchronous Load of JavaScript in HTML
	HTML: HTTP 1.0 Usage



หมายถึง ผ่านการทดสอบ



หมายถึง ไม่ผ่านทดสอบ



หมายถึง ผ่านการทดสอบแต่มีคำแนะนำ



หมายถึง มีแจ้งเตือนแต่ยังผ่านการทดสอบ

Duplicate Content

App ไม่ควร Download ข้อมูลที่ซ้ำซ้อน (กรณีที่ Download มาแล้ว) หากข้อมูลไม่ถูกเปลี่ยนแปลง

วิธีการทดสอบ : รับภาพมาหนึ่งภาพ เปิด 1 ครั้ง ปิดภาพ ออกจากแชท กลับเข้ามาแล้วเปิดอีกครั้ง

ครั้งที่	ผลลัพธ์	รายละเอียด
1	Signal App ไม่ Download ภาพซ้ำ เนื่องจาก Signal ได้เก็บภาพไว้ใน Cache (Database) (ทดลองโดย ผู้ทดสอบปิด Internet ขณะดูภาพเป็นครั้งที่ 2)	-
2	Signal App ไม่ Download ภาพซ้ำ เนื่องจาก Signal ได้เก็บภาพไว้ใน Cache (Database) (ทดลองโดย ผู้ทดสอบปิด Internet ขณะดูภาพเป็นครั้งที่ 2)	-
3	Signal App ไม่ Download ภาพซ้ำ เนื่องจาก Signal ได้เก็บภาพไว้ใน Cache (Database) (ทดลองโดย ผู้ทดสอบปิด Internet ขณะดูภาพเป็นครั้งที่ 2)	-
สรุป		ผ่าน

Cache Control

ทดสอบการจัดการด้าน Cache ว่าภายใน Signal มีการจัดการ Cache หรือไม่

วิธีการทดสอบ : ส่งข้อความและภาพไปยัง App ที่ทำงานบนสมาร์ตโฟนอีกเครื่องจากนั้นทำการปิด App แล้วเปิดใหม่ ดูว่า มี Cache อยู่หรือไม่

ครั้งที่	ผลลัพธ์	รายละเอียด
1	มีการเก็บข้อความและภาพต่าง ๆ ไว้ในเครื่อง เมื่อปิด App และเปิดขึ้นมาใหม่โดยไม่เชื่อมต่อ กับ Internet App สามารถแสดงข้อความและภาพได้	-
2	มีการเก็บข้อความและภาพต่าง ๆ ไว้ในเครื่อง เมื่อปิด App และเปิดขึ้นมาใหม่โดยไม่เชื่อมต่อ กับ Internet App สามารถแสดงข้อความและภาพได้	-
3	มีการเก็บข้อความและภาพต่าง ๆ ไว้ในเครื่อง เมื่อปิด App และเปิดขึ้นมาใหม่โดยไม่เชื่อมต่อ กับ Internet App สามารถแสดงข้อความและภาพได้	-
สรุป		ผ่าน

Cache Expiration

การทดสอบนี้จะตรวจสอบอายุของ Cache ว่ามีวันหมดอายุหรือไม่ (Cache ควรจะมีวันหมดอายุ)

วิธีการทดสอบ : ดูว่า message หรือรูปใน Cache มีอายุของ Cache หรือไม่ ในลักษณะคล้าย ๆ กับแอปพลิเคชัน Line

ครั้งที่	ผลลัพธ์	รายละเอียด
1	ไม่มีอายุของ Cache เนื่องจากเก็บข้อมูลต่าง ๆ ลงใน database ของเครื่องจนกว่าผู้ใช้จะลบเอง (หลังจาก ผู้ทดลองรอ 2 อาทิตย์พบว่า ข้อมูลดังกล่าวก็ยังอยู่ใน Database)	-
2	ไม่มีอายุของ Cache เนื่องจากเก็บข้อมูลต่าง ๆ ลงใน database ของเครื่องจนกว่าผู้ใช้จะลบเอง (หลังจาก ผู้ทดลองรอ 2 อาทิตย์พบว่า ข้อมูลดังกล่าวก็ยังอยู่ใน Database)	-
3	ไม่มีอายุของ Cache เนื่องจากเก็บข้อมูลต่าง ๆ ลงใน database ของเครื่องจนกว่าผู้ใช้จะลบเอง (หลังจาก ผู้ทดลองรอ 2 อาทิตย์พบว่า ข้อมูลดังกล่าวก็ยังอยู่ใน Database)	-
สรุป		ไม่ผ่าน

Image Metadata

Signal ควรจะต้องมีการตัด Metadata ที่ไม่จำเป็นออกไปก่อนทำการส่ง

วิธีการทดสอบ : ส่งภาพเพื่อตรวจสอบว่า Metadata ถูกตัดออกก่อนส่งหรือไม่

ครั้งที่	ผลลัพธ์	รายละเอียด
1	มีการตัด metadata ก่อนส่ง เช่น ชื่อ เวลาที่ถ่ายภาพ สถานที่ถ่ายภาพ	-
2	มีการตัด metadata ก่อนส่ง เช่น ชื่อ เวลาที่ถ่ายภาพ สถานที่ถ่ายภาพ	-
3	มีการตัด metadata ก่อนส่ง เช่น ชื่อ เวลาที่ถ่ายภาพ สถานที่ถ่ายภาพ	-
สรุป		ผ่าน

Opening Connections

การเปิด Connection ไม่ควรมีมากเกินไป Signal ควรจะจัดกลุ่มข้อมูลก่อนแล้วจึงส่งข้อมูลไปใน Connection เดียว (ในหัวข้อ 4.1.10)

วิธีการทดสอบ : วิเคราะห์จาก Trace results ใน ARO

ครั้งที่	ผลลัพธ์	รายละเอียด
1	มีเพียงคำแนะนำ ไม่พบการแจ้งเตือนสำหรับการแก้ไข	-
2	มีเพียงคำแนะนำ ไม่พบการแจ้งเตือนสำหรับการแก้ไข	-
3	มีเพียงคำแนะนำ ไม่พบการแจ้งเตือนสำหรับการแก้ไข	-
สรุป		ผ่าน(พร้อมคำแนะนำ)

คำแนะนำ : ถ้าหากพบว่า App เปิด Connection พร้อม ๆ กันจำนวนมาก ให้พิจารณา
 ที่ทำการรวม Connection เข้าไว้ด้วยกัน

Multiple Simultaneous TCP Connections

การเปิด Connection พร้อมกันมากเกินไปจนความจำเป็น Signal ควรจัดกลุ่มก่อนแล้วส่งไปใน Connection เดียว

วิธีการทดสอบ : วิเคราะห์จาก Trace results ใน ARO ในขณะที่เปิด Chat กลุ่ม คุยกันหลาย ๆ คน load รูปพร้อม ๆ กันหลาย ๆ รูป

ครั้งที่	ผลลัพธ์	รายละเอียด
1	พบการแจ้งเตือนของ IP Signal 1 ครั้ง	มีการส่ง packet 19 packet ไปที่ปลายทาง แห่งเดียว โดย Signal ไม่ได้ส่งข้อความ และ รูปภาพต่าง ๆ ไปพร้อมกัน (เว้นระยะการส่ง)
2	พบการแจ้งเตือนของ IP Signal 2 ครั้ง	มีการส่ง packet 19 packet ไป แต่เป็นการ ส่งข้อความ รูปภาพไปที่ปลายทางหลายแห่ง จึงทำให้ทาง app ไม่สามารถตัดสินใจส่งรวมไป ที่เดียวได้
3	ไม่พบการแจ้งเตือนของ app จาก ARO	เนื่องจาก network มีปัญหาขณะส่งข้อมูล ทำให้มีการรวบรวมข้อมูลส่งไปด้วยกันทีเดียวจึง ทำให้ไม่เกิดการแจ้งเตือน
สรุป		ไม่ผ่าน (เนื่องจาก ครั้งที่ 1 และ 2)

คำอธิบาย: ในกรณีที่สัญญาณการเชื่อมต่อดี Signal พยายามส่งข้อความทันที โดยไม่ Bundle แต่ จากการทดลองครั้งที่ 3 พบว่า สัญญาณการเชื่อมต่อมีปัญหา Signal จะ Bundle ข้อความแล้วส่งทีเดียว

Periodic Transfers

ตรวจสอบว่า Signal มีการเชื่อมต่อเพื่อส่ง Log ต่าง ๆ หรือ Ad ที่เป็นโฆษณาเป็นระยะ ๆ หรือไม่

วิธีการทดสอบ : วิเคราะห์จาก Trace results ใน ARO

ครั้งที่	ผลลัพธ์	รายละเอียด
1	ไม่พบการแจ้งเตือนใด ๆ จาก ARO	ARO ไม่พบการแจ้งเตือน ใด ๆ แสดงว่าไม่มีการเก็บ log หรือ ad ต่าง ๆ (ที่ต้องส่งข้อมูลเป็นระยะ ๆ)
2	ไม่พบการแจ้งเตือนใด ๆ จาก ARO	ARO ไม่พบการแจ้งเตือน ใด ๆ แสดงว่าไม่มีการเก็บ log หรือ ad ต่าง ๆ (ที่ต้องส่งข้อมูลเป็นระยะ ๆ)
3	ไม่พบการแจ้งเตือนใด ๆ จาก ARO	ARO ไม่พบการแจ้งเตือน ใด ๆ แสดงว่าไม่มีการเก็บ log หรือ ad ต่าง ๆ (ที่ต้องส่งข้อมูลเป็นระยะ ๆ)
สรุป		ผ่าน

Screen Rotations

การหมุนหน้าจอของ Signal หนีเนื้อหาเดิม ไม่ควรจะมีการเชื่อมต่อกับ Server เพื่อรับข้อมูลอีก

วิธีการทดสอบ : ลอง Rotate Screen ดู แล้วดูว่า มีการ Load เพิ่มหรือไม่

ครั้งที่	ผลลัพธ์	รายละเอียด
1	ไม่มีการโหลดข้อมูลใหม่ และไม่มีการ open connection ใด ๆ	-
2	ไม่มีการโหลดข้อมูลใหม่ และไม่มีการ open connection ใด ๆ	-
3	ไม่มีการโหลดข้อมูลใหม่ และไม่มีการ open connection ใด ๆ	-
สรุป		ผ่าน

Closing Connections

ตรวจสอบว่า Signal ปิด Connection ของ Signal ว่าเมื่อไม่ใช้งานแล้วหรือไม่

วิธีการทดสอบ : ตรวจสอบดูว่า มี Connection เปิดค้างหรือไม่

ครั้งที่	ผลลัพธ์	รายละเอียด
1	ไม่พบการเปิด connection ทิ้งไว้	ผ่านการตรวจสอบ ARO ไม่พบการเปิด connection ใด ๆ ค้างไว้
2	ไม่พบการเปิด connection ทิ้งไว้	ผ่านการตรวจสอบ ARO ไม่พบการเปิด connection ใด ๆ ค้างไว้
3	ไม่พบการเปิด connection ทิ้งไว้	ผ่านการตรวจสอบ ARO ไม่พบการเปิด connection ใด ๆ ค้างไว้
สรุป		ผ่าน

Offloading to Wi-Fi

ตรวจสอบว่า Signal สามารถสลับไปมาระหว่าง Wi-Fi กับ 3G ได้หรือไม่ หากมีทั้ง 3G/4G และ Wi-Fi Signal ควรเปลี่ยนไปใช้ Wi-Fi

วิธีการทดสอบ : ทดลองใช้ 3G ก่อน จากนั้น ลองเปิด Wi-Fi ตรวจสอบว่า app ทำการ switch มาใช้ Wi-Fi หรือไม่

ครั้งที่	ผลลัพธ์	รายละเอียด
1	app ทำการ switch จาก 3G/LTE มาเป็น Wi-Fi	-
2	app ทำการ switch จาก 3G/LTE มาเป็น Wi-Fi	-
3	app ทำการ switch จาก 3G/LTE มาเป็น Wi-Fi	-
สรุป		ผ่าน

HTTP 400 and 500 Status Codes

ตรวจสอบว่า Signal มีการเรียกใช้บริการ Web ที่ Return Code HTTP 400 หรือ 500 หรือไม่

วิธีการทดสอบ : วิเคราะห์จาก Trace results ใน ARO

ครั้งที่	ผลลัพธ์	รายละเอียด
1	ไม่มีการตรวจพบ	ARO ไม่พบการเชื่อมต่อใด ๆ ที่ทำให้เกิด code 400 - 500
2	ไม่มีการตรวจพบ	ARO ไม่พบการเชื่อมต่อใด ๆ ที่ทำให้เกิด code 400 - 500
3	ไม่มีการตรวจพบ	ARO ไม่พบการเชื่อมต่อใด ๆ ที่ทำให้เกิด code 400 - 500
สรุป		ผ่าน

HTTP 300 Status Codes

ตรวจสอบว่า Signal มีการเรียกใช้บริการ Web ที่ Return Code HTTP 300 หรือไม่

วิธีการทดสอบ : วิเคราะห์จาก Trace results ใน ARO

ครั้งที่	ผลลัพธ์	รายละเอียด
1	ไม่มีการตรวจพบ	ARO ไม่พบการเชื่อมต่อใด ๆ ที่ทำให้เกิด code 300
2	ไม่มีการตรวจพบ	ARO ไม่พบการเชื่อมต่อใด ๆ ที่ทำให้เกิด code 300
3	ไม่มีการตรวจพบ	ARO ไม่พบการเชื่อมต่อใด ๆ ที่ทำให้เกิด code 300
สรุป		ผ่าน

HTTP 1.0 Usage

ตรวจสอบว่า Signal มีการเรียกใช้ API หรือการเชื่อมต่อไปยัง HTTP 1.0 หรือไม่ ซึ่งไม่ควรมี

วิธีการทดสอบ : วิเคราะห์จาก Trace results ใน ARO

ครั้งที่	ผลลัพธ์	รายละเอียด
1	ไม่มีการตรวจพบ	ARO ตรวจไม่พบการส่งข้อมูลแบบ HTTP 1.0
2	ไม่มีการตรวจพบ	ARO ตรวจไม่พบการส่งข้อมูลแบบ HTTP 1.0
3	ไม่มีการตรวจพบ	ARO ตรวจไม่พบการส่งข้อมูลแบบ HTTP 1.0
สรุป		ผ่าน

Unsecure SSL Version

ตรวจสอบว่า Signal มีการใช้ SSL Version ที่ถูกระบุว่าไม่ปลอดภัยหรือไม่

วิธีการทดสอบ : วิเคราะห์จาก Trace results ใน ARO

ครั้งที่	ผลลัพธ์	รายละเอียด
1	ไม่มีการตรวจพบ	ARO ตรวจไม่พบ Unsecure SSL Version
2	ไม่มีการตรวจพบ	ARO ตรวจไม่พบ Unsecure SSL Version
3	ไม่มีการตรวจพบ	ARO ตรวจไม่พบ Unsecure SSL Version
สรุป		ผ่าน

Weak Cipher

ตรวจสอบว่า Signal มีการเข้ารหัส SSL ที่มีความปลอดภัยต่ำ ถูกถอดรหัสได้ง่ายหรือไม่

วิธีการทดสอบ : วิเคราะห์จาก Trace results ใน ARO

ครั้งที่	ผลลัพธ์	รายละเอียด
1	ไม่มีการตรวจพบ	ARO ตรวจไม่พบ weak cipher
2	ไม่มีการตรวจพบ	ARO ตรวจไม่พบ weak cipher
3	ไม่มีการตรวจพบ	ARO ตรวจไม่พบ weak cipher
สรุป		ผ่าน

Forward Secrecy

ตรวจสอบความปลอดภัยของ Signal ว่าป้องกันเรื่อง Forward Secrecy ซึ่งหาก Signal มีการป้องกันแล้ว เหตุการณ์การถูกถอดรหัส Session หนึ่งได้แล้วจะทำให้เข้าไปในอีก Session ก่อนหน้านั้น ทั้งหมดได้ จะไม่เกิดขึ้น

วิธีการทดสอบ : วิเคราะห์จาก Trace results ใน ARO

ตารางที่ 419 Forward Secrecy

ครั้งที่	ผลลัพธ์	รายละเอียด
1	ไม่มีการตรวจพบ	ARO ตรวจไม่พบ support forward
2	ไม่มีการตรวจพบ	ARO ตรวจไม่พบ support forward
3	ไม่มีการตรวจพบ	ARO ตรวจไม่พบ support forward
สรุป		ผ่าน

HTTP Vs. HTTPS

ตรวจสอบว่า API ต่าง ๆ ของ Signal ใช้ HTTPS ทั้งหมดหรือไม่

วิธีการทดสอบ : วิเคราะห์จาก Trace results ใน ARO

ครั้งที่	ผลลัพธ์	รายละเอียด
1	ไม่มีการตรวจพบ	ARO ตรวจไม่พบ HTTP
2	ไม่มีการตรวจพบ	ARO ตรวจไม่พบ HTTP
3	ไม่มีการตรวจพบ	ARO ตรวจไม่พบ HTTP
สรุป		ผ่าน

Private Data

ตรวจสอบว่า Signal ได้ขอข้อมูลส่วนตัวอะไรบ้างจากผู้ใช้ และมีการป้องกันข้อมูลเหล่านั้นอย่างไรเมื่อมีการส่งไปยัง Server

วิธีการทดสอบ : นอกจากการใช้ ARO ทดสอบผู้จัดทำโครงการนี้ได้ตรวจสอบจาก manifest ของ android และได้สังเกตขั้นตอน Register Signal ว่าเข้ารหัสข้อมูลก่อนส่งหรือไม่

ครั้งที่	ผลลัพธ์	รายละเอียด
1	มีการขอ data ต่าง ๆ ดังนี้ profile, เบอร์โทรศัพท์, การเข้าถึง log, การขอใช้พื้นที่ภายในเครื่อง, การขอที่อยู่, การอ่าน SMS, การสมัครมีการส่งข้อมูล เบอร์โทรเป็นแบบเปิดเผย	-
2	มีการขอ data ต่าง ๆ ดังนี้ profile, เบอร์โทรศัพท์, การเข้าถึง log, การขอใช้พื้นที่ภายในเครื่อง, การขอที่อยู่, การอ่าน SMS, การสมัครมีการส่งข้อมูล เบอร์โทรเป็นแบบเปิดเผย	-
3	มีการขอ data ต่าง ๆ ดังนี้ profile, เบอร์โทรศัพท์, การเข้าถึง log, การขอใช้พื้นที่ภายในเครื่อง, การขอที่อยู่, การอ่าน SMS, การสมัครมีการส่งข้อมูล เบอร์โทรเป็นแบบเปิดเผย	-
สรุป		ไม่ผ่าน

ผลจากการตรวจสอบไฟล์ AndroidManifest จากการสังเกตขั้นตอน Registration จาก Log ของ Server แสดงให้เห็นว่าหมายเลขโทรศัพท์ไม่มีการเข้ารหัส

Accessing Peripherals

ตรวจสอบการขอใช้ Hardware ต่าง ๆ ของ Signal ว่าเมื่อมีการใช้งาน หลังจากเสร็จสิ้นการใช้งาน Signal ทำการปิดอุปกรณ์เหล่านั้นให้ด้วยหรือไม่

วิธีการทดสอบ : ตรวจสอบใน manifest ว่าควรปิด GPS, Bluetooth ฯลฯ เมื่อไม่ใช้งานแล้ว

ครั้งที่	ผลลัพธ์	รายละเอียด
1	มีการขอใช้ GPS Bluetooth แต่ทั้งนี้ทั้งนั้น ทาง signal จะไม่เปิดใช้เอง หากผู้ใช้ประสงค์จะใช้งาน signal จึงจะขออนุญาต ในการเปิดอีกครั้ง และหากจะปิดผู้ใช้ต้องปิดที่ android settings ด้วยตัวเอง เนื่องจาก signal โดยการทำงานให้ android	-
2	มีการขอใช้ GPS Bluetooth แต่ทั้งนี้ทั้งนั้น ทาง signal จะไม่เปิดใช้เอง หากผู้ใช้ประสงค์จะใช้งาน signal จึงจะขออนุญาต ในการเปิดอีกครั้ง และหากจะปิดผู้ใช้ต้องปิดที่ android settings ด้วยตัวเอง เนื่องจาก signal โดยการทำงานให้ android	-
3	มีการขอใช้ GPS Bluetooth แต่ทั้งนี้ทั้งนั้น ทาง signal จะไม่เปิดใช้เอง หากผู้ใช้ประสงค์จะใช้งาน signal จึงจะขออนุญาต ในการเปิดอีกครั้ง และหากจะปิดผู้ใช้ต้องปิดที่ android settings ด้วยตัวเอง เนื่องจาก signal โดยการทำงานให้ android	-
สรุป		ไม่ผ่าน

ไม่ผ่านเนื่องจาก Signal ไม่ปิด GPS ให้อัตโนมัติ ขณะไม่ใช้งาน

การทดสอบประสิทธิภาพของ Signal

เลือกทดสอบ Performance Testing โดยใช้ JMeter ตามหัวข้อดังนี้

- Load Testing
- Stress Testing
- Capacity Testing

ผลการดำเนินโครงการจะทำการสรุปผลจากการทดสอบประสิทธิภาพจำนวน 3 ครั้ง หากมีครั้งใดที่พบข้อผิดพลาด ผลการทดสอบนั้นจะถือว่าไม่สามารถรองรับจำนวนผู้ใช้งานตามหัวข้อการทดสอบประสิทธิภาพนั้นได้ ยกเว้นหัวข้อ Capacity Testing ที่จะนำผลการทดสอบจำนวน 3 ครั้งที่ได้มาหาค่าเฉลี่ย

การทดสอบระบบแบบ Load Testing จะทดสอบการทำงานของระบบเมื่อต้องให้บริการคำร้องขอ 50 คำร้องขอภายในเวลา 1 วินาที

การทดสอบระบบแบบ Stress Testing จะทดสอบการทำงานของระบบเมื่อต้องให้บริการคำร้องขอมากกว่าปกติ มีคำร้องขอ 100 คำร้องขอ ภายในเวลา 1 วินาที

การทดสอบแบบ Capacity Testing จะทดสอบการทำงานของระบบ เมื่อต้องให้บริการคำร้องขอสูงสุดที่สามารถร้องขอได้ ภายในระยะเวลา 1 วินาที

API สำหรับการทดสอบประสิทธิภาพมีดังต่อไปนี้

- GET /v1/directory/{tokens}
- GET /v1/profile/{number}
- GET /v2/keys/{number}/{device_id}
- PUT /v1/messages/{destination_number}
- GET /v1/attachments/
- GET /v1/attachments/{attachment_id}

สรุปผลการทดสอบประสิทธิภาพหัวข้อ Load Testing และ Stress Testing ภายในระยะเวลา 1 วินาที

หัวข้อการทดสอบประสิทธิภาพ	ค่าเฉลี่ยระยะเวลา ตอบสนอง (มิลลิ วินาที) ต่อจำนวนผู้ ใช้งาน	ผลการทดสอบ ประสิทธิภาพ
1. Load Testing		
1.1 GET /v1/directory/{tokens}	1807.6	สามารถรองรับได้
1.2 GET /v1/profile/{number}	1591.3	สามารถรองรับได้
1.3 GET /v2/keys/{number}/{drvice_id}	3297	ไม่สามารถรองรับได้
1.4 PUT /v1/messages /{destination_number}	3168.6	สามารถรองรับได้
1.5 GET /v1/attachments/	2160.6	สามารถรองรับได้
1.6 GET /v1/attachments/{attachment_id}	1510	สามารถรองรับได้
2. Stress Testing		
2.1 GET /v1/directory/{tokens}	3419	สามารถรองรับได้
2.2 GET /v1/profile/{number}	2425	สามารถรองรับได้
2.3 GET /v2/keys/{number}/{drvice_id}	3984.6	ไม่สามารถรองรับได้
2.4 PUT /v1/messages /{destination_number}	5202.6	ไม่สามารถรองรับได้
2.5 GET /v1/attachments/	3614	ไม่สามารถรองรับได้
2.6 GET /v1/attachments/{attachment_id}	2212.3	สามารถรองรับได้

สรุปผลการทดสอบประสิทธิภาพหัวข้อ Capacity Testing ภายในระยะเวลา 1 วินาที

หัวข้อการทดสอบประสิทธิภาพ	ค่าเฉลี่ยจำนวนผู้ใช้งานสูงสุดที่ เครื่องแม่ข่ายยังสามารถรองรับได้ (คนต่อ 1 วินาที)
3. Capacity Testing	
1. GET /v1/directory/{tokens}	1230
2. GET /v1/profile/{number}	1058
3. GET /v2/keys/{number}/{drvice_id}	1
4. PUT /v1/messages/{destination_number}	82
5. GET /v1/attachments/	92
6. GET /v1/attachments/{attachment_id}	1195

บทที่ 3

อภิปรายผลการวิจัย

เราได้ทำการเปรียบเทียบ messaging app ทั้ง 6 ตัวในแง่มุมต่อไปนี้

- การเข้ารหัสในช่องส่งข้อความ
- ผู้ให้บริการไม่สามารถถอดรหัสเพื่อดูบทสนทนาได้
- ยืนยันตัวตนของผู้ติดต่อด้วยได้หรือไม่
- ถ้าถูกแจ้งถอดรหัสหาย ข้อมูลที่มีมาในอดีตจะปลอดภัย
- โปรแกรมเปิดเผยซอร์สเพื่อให้ประเมิน
- การออกแบบความปลอดภัยระบบมีความชัดเจน มีเอกสารกำกับทุกขั้นตอน
- มีการตรวจสอบซอร์สโค้ดล่าสุด

เราสามารถจะสรุปคุณสมบัติของ messaging app แต่ละตัวได้ตามตารางดังต่อไปนี้

	การเข้ารหัสในช่องส่งข้อความ	ผู้ให้บริการไม่สามารถถอดรหัสเพื่อดูบทสนทนาได้	ยืนยันตัวตนของผู้ติดต่อด้วยได้	ถ้าถูกแจ้งถอดรหัสหาย ข้อมูลที่มีมาในอดีตจะปลอดภัย	โปรแกรมเปิดเผยซอร์สเพื่อให้ประเมิน	การออกแบบความปลอดภัยระบบมีความชัดเจน มีเอกสารกำกับทุกขั้นตอน	มีการตรวจสอบซอร์สโค้ดล่าสุด
LINE	✓	✗	✓	✗	✗	✓	✓
Skype	✓	✗	✗	✗	✗	✗	✗
WhatsApp	✓	✓	✓	✓	✗	✓	✓
Telegram	✓	✗	✗	✗	✓	✓	✓
Signal	✓	✓	✓	✓	✓	✓	✓
Snapchat	✓	✗	✗	✗	✗	✗	✓

โดยคณะผู้วิจัยเลือกจะพัฒนาต่อจาก codebase ของ Signal เพราะมีคุณสมบัติทั้ง 6 ครบถ้วน แตกต่างจาก messaging app อื่นๆที่พิจารณาที่ขาดคุณสมบัติบางตัวไป

หลังจากเราได้ดัดแปลง ติดตั้ง และทดสอบ Signal ที่เซิร์ฟเวอร์ของ DigitalOcean ซึ่งเป็นผู้ให้บริการคลาวด์รายใหญ่แล้ว เราได้ทำสิ่งต่อไปนี้สำเร็จ

1. สามารถติดตั้ง Signal Server และใช้งาน Signal Android messaging application ผ่าน Signal Server ที่ติดตั้งเองได้
2. สามารถใช้งานโปรแกรม Application Resource Optimizer (ARO) จาก AT&T เพื่อทดสอบ Signal ตาม “Best Practices” ของ ARO ได้ ผลการทดสอบชี้ให้เห็นว่า Signal ผ่านเกณฑ์ทางด้านความปลอดภัย

และความเป็นส่วนตัวเกือบทั้งหมด โดยเกณฑ์ที่ไม่ผ่านนั้นเราสามารถจะปรับแต่ง source code เพื่อให้เป็นไปตามเกณฑ์ที่กำหนดได้

3. สามารถจัดการกับการส่งข้อมูลมีเดียอย่างปลอดภัยได้ โดยใช้บริการของ open-source cloud storage service คือ Minio

4. สามารถทดสอบประสิทธิภาพการรองรับของผู้ใช้งานจำนวนมากและได้ผลลัพธ์เป็นที่น่าพอใจ ระบบสามารถรองรับผู้ใช้งานได้เฉลี่ย 50 คนต่อวินาทีในการติดตั้งแบบเซิร์ฟเวอร์เดี่ยว และระบบสามารถทำการขยายในทางราบ (horizontal scaling) โดยการเพิ่มเซิร์ฟเวอร์ เพื่อรองรับผู้ใช้งานที่เพิ่มขึ้นต่อวินาทีได้

บทที่ 4

สรุปและข้อเสนอแนะ

งานวิจัยนี้ได้บรรลุวัตถุประสงค์ในเฟสที่หนึ่งกล่าวคือ

- ได้คัดเลือกโครงการโอเพ่นซอร์สสำหรับสร้างแอปพลิเคชันการรับส่งข้อมูลที่ต้องการความปลอดภัยสูงที่เหมาะสมคือ Signal เพื่อนำมาใช้เป็นฐานในการพัฒนา ดัดแปลง ให้เหมาะกับการใช้งานและสภาพแวดล้อมของหน่วยงานของรัฐในประเทศไทยที่ต้องการใช้งานแอปพลิเคชันในลักษณะนี้เช่น กรมข่าวทหารบก เป็นต้น
- ได้ทำการศึกษา ดัดแปลง และติดตั้ง ส่วนเซิร์ฟเวอร์ของแอปพลิเคชัน Signal บนคลาวด์ พร้อมทั้งได้ทดสอบการใช้งานแอปพลิเคชันผ่านเซิร์ฟเวอร์นี้

ในเฟสต่อไปเราจะได้ทำการปรับปรุงซอร์สโค้ดของ Signal เพื่อกำจัดข้อบกพร่องของแอปพลิเคชันที่ได้เรียนรู้จากการทดสอบ นอกจากนั้นเราจะขยายการรองรับผู้ใช้งานต่อวินาทีให้มีจำนวนมากขึ้น และทำการทดสอบระบบกับผู้ใช้งานจริงจำนวนมาก จนเมื่อได้แอปพลิเคชันที่มีความเสถียรและมีสมรรถนะดีแล้ว เราจะส่งเสริมให้มีการใช้แอปพลิเคชันนี้ในองค์กรของรัฐที่ต้องจัดการกับความมั่นคงของข้อมูลอย่างแพร่หลาย

บทที่ 5

ผลผลิต

โปรแกรมที่ได้ดัดแปลงและพัฒนาต่อจากฐานซอร์สโค้ดของ Signal ซึ่งสามารถดาวน์โหลดหรือ clone ได้จากลิงค์ต่อไปนี่:

- <https://github.com/SeksunNiampan/docker-Signal-Server>
- <https://github.com/SeksunNiampan/Signal-Server>
- <https://github.com/SeksunNiampan/Signal-Android>

บรรณานุกรม

[1] ARO [ออนไลน์] เข้าถึงได้จาก

<https://developer.att.com/application-resource-optimizer> (วันที่ค้นข้อมูล: 10 มกราคม 2560)

[2] ARO Best-Practices [ออนไลน์] เข้าถึงได้จาก

<https://developer.att.com/application-resource-optimizer/docs/best-practices> (วันที่ค้นข้อมูล: 10 มกราคม 2560)

[3] ATT ARO Analysis Guide [ออนไลน์] เข้าถึงได้จาก

https://developer.att.com/static-assets/documents/aro/ATT_ARO_Analysis_Guide_4-0.pdf (วันที่ค้นข้อมูล: 10 มกราคม 2560)

[4] PC Mag Only 6 Messaging Apps Are Truly Secure [ออนไลน์] เข้าถึงได้จาก

<http://www.pcmag.com/article2/0,2817,2471658,00.asp> (วันที่ค้นข้อมูล: 10 มกราคม 2560)

[5] Signal Android [ออนไลน์] เข้าถึงได้จาก

<https://github.com/WhisperSystems/Signal-Android/wiki> (วันที่ค้นข้อมูล: 10 มกราคม 2560)

[6] Signal News Topics [ออนไลน์] เข้าถึงได้จาก

<https://www.blognone.com/topics/signal> (วันที่ค้นข้อมูล: 20 เมษายน 2560)

[7] Signal Setup Server [ออนไลน์] เข้าถึงได้จาก

<https://github.com/lucaconte/BeatTheMeddler/blob/a16faea568c41150c10eb4162b74b94faebfb7/README.md> (วันที่ค้นข้อมูล: 10 มกราคม 2560)

[8] Signal VS WhatApps [ออนไลน์] เข้าถึงได้จาก

<https://www.blognone.com/node/81380> (วันที่ค้นข้อมูล: 20 เมษายน 2560)

[9] Android Fingerprint Authentication [ออนไลน์] เข้าถึงได้จาก

http://www.techotopia.com/index.php/An_Android_Fingerprint_Authentication_Tutorial (วันที่ค้นข้อมูล: 24 พฤษภาคม 2560)

[10] Fingerprint API Tutorial [ออนไลน์] เข้าถึงได้จาก

<http://wanttobeanandroiddev.blogspot.com/2016/04/android-fingerprint-api-tutorial.html> (วันที่ค้นข้อมูล: 24 พฤษภาคม 2560)

- [11] Activity Lifecycle พื้นฐาน Android ที่ Developer ควรรู้ [ออนไลน์] เข้าถึงได้จาก
<http://www.artit-k.com/android-activity-lifecycle/> (วันที่ค้นข้อมูล: 24 พฤษภาคม 2560)
- [12] เก็บค่าตัวแปรให้ถาวรแบบง่ายๆได้ด้วย Shared Preferences [ออนไลน์] เข้าถึงได้จาก
<http://www.akexorcist.com/2014/09/android-shared-preferences.html> (วันที่ค้นข้อมูล: 24 พฤษภาคม 2560)
- [13] LockPattern [ออนไลน์] เข้าถึงได้จาก
<https://github.com/pro100svitlo/LockPattern> (วันที่ค้นข้อมูล: 24 พฤษภาคม 2560)
- [14] docker [ออนไลน์] เข้าถึงได้จาก
<https://docs.docker.com/> (วันที่ค้นข้อมูล: 5 มีนาคม 2561)
- [15] docker [ออนไลน์] เข้าถึงได้จาก
<https://arnondora.in.th/whats-docker> (วันที่ค้นข้อมูล: 5 มีนาคม 2561)
- [16] Install java maven [ออนไลน์] เข้าถึงได้จาก
<http://www.dangtrinh.com/2014/04/install-and-configure-java-and-maven-in.html> (วันที่ค้นข้อมูล: 8 มีนาคม 2561)
- [17] Install maven 3.5.2 [ออนไลน์] เข้าถึงได้จาก
<https://tecadmin.net/install-apache-maven-on-ubuntu/> (วันที่ค้นข้อมูล: 9 มีนาคม 2561)
- [18] docker postgres environment [ออนไลน์] เข้าถึงได้จาก
<https://github.com/docker-library/docs/tree/master/postgres> (วันที่ค้นข้อมูล: 21 มีนาคม 2561)
- [19] Minio [ออนไลน์] เข้าถึงได้จาก
<https://www.Minio.io/> (วันที่ค้นข้อมูล: 26 มีนาคม 2561)
- [20] docker Minio [ออนไลน์] เข้าถึงได้จาก
<https://hub.docker.com/r/Minio/Minio/> (วันที่ค้นข้อมูล: 26 มีนาคม 2561)
- [21] Minio docs [ออนไลน์] เข้าถึงได้จาก
<https://docs.Minio.io/> (วันที่ค้นข้อมูล: 28 มีนาคม 2561)
- [22] Minio docker compose [ออนไลน์] เข้าถึงได้จาก

<https://github.com/Minio/Minio/blob/master/docs/orchestration/docker-swarm/docker-compose.yaml> (วันที่ค้นข้อมูล: 28 มีนาคม 2561)

[23] Linux install docker [ออนไลน์] เข้าถึงได้จาก

<https://docs.docker.com/install/linux/docker-ce/ubuntu/#prerequisites> (วันที่ค้นข้อมูล: 9 เมษายน 2561)

[24] Signal api protocol [ออนไลน์] เข้าถึงได้จาก

<https://github.com/signalapp/Signal-Server/wiki/API-Protocol> (วันที่ค้นข้อมูล: 18 เมษายน 2561)

[25] Performance Testing [ออนไลน์] เข้าถึงได้จาก

<https://www.aware.co.th/software-performance-testing/> (วันที่ค้นข้อมูล: 26 เมษายน 2561)

[26] Performance Testing [ออนไลน์] เข้าถึงได้จาก

<http://www.howtoautomate.in.th/theory-web-performance-testing-101/> (วันที่ค้นข้อมูล: 26 เมษายน 2561)

[27] Performance Testing [ออนไลน์] เข้าถึงได้จาก

<https://blog.smartbear.com/software-quality/7-types-of-web-performance-tests-and-how-they-fit-into-your-testing-cycle/> (วันที่ค้นข้อมูล: 27 เมษายน 2561)

[28] Summary report Jmeter [ออนไลน์] เข้าถึงได้จาก

<http://www.testingjournals.com/understand-summary-report-jmeter/> (วันที่ค้นข้อมูล: 7 พฤษภาคม 2561)

[29] Jmeter plugin [ออนไลน์] เข้าถึงได้จาก

<https://jmeter-plugins.org/> (วันที่ค้นข้อมูล: 7 พฤษภาคม 2561)

[30] Jmeter load test [ออนไลน์] เข้าถึงได้จาก

<https://sysadmin.psu.ac.th/2017/02/10/jmeter-01-basic-test-plan/> (วันที่ค้นข้อมูล: 15 พฤษภาคม 2561)