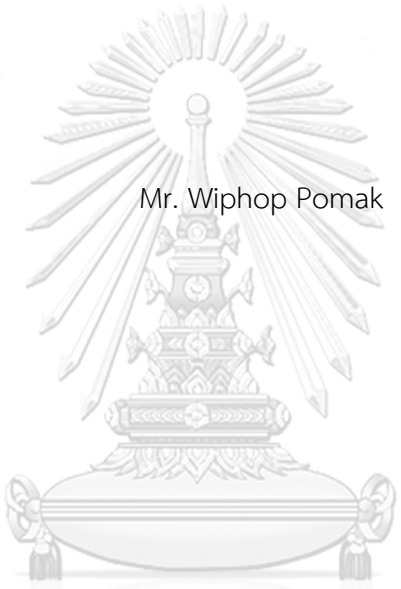
จุดพร้อมโยงที่ปลอดภัยด้วยการพิสูจน์ตัวจริงแบบหลายปัจจัยผ่านเอ็นเอฟซีบนสมาร์ตโฟน

นายวิภพ โพธิ์มาก

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2560

SECURE HOTSPOTS WITH MULTI-FACTOR AUTHENTICATION THROUGH NFC ON SMARTPHONES

Mr. Wiphop Pomak



A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Computer Science
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2017

หัวข้อวิทยานิพนธ์	จุดพร้อมโยงที่ปลอดภัยด้วยการพิสูจน์ตัวจริงแบบหลายปัจจัยผ่านเอ็นเอฟซีบนสมาร์ตโฟน
โดย	นายวิภพ โพธิ์มาก
สาขาวิชา	วิทยาศาสตร์คอมพิวเตอร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	รองศาสตราจารย์ ดร.ญาใจ ลิ่มปิยะกรณ์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

.....คณบดีคณะวิศวกรรมศาสตร์

(รองศาสตราจารย์ ดร.สุพจน์ เตชวรสินสกุล)

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.สุกรี สินธุภิญโญ)

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

(รองศาสตราจารย์ ดร.ญาใจ ลิ่มปิยะกรณ์)

.....กรรมการภายนอกมหาวิทยาลัย

(อาจารย์ ดร.ภาสกร อภิรักษ์วรพินิต)

วิภพ โพธิ์มาก : จุดพร้อมโยงที่ปลอดภัยด้วยการพิสูจน์ตัวจริงแบบหลายปัจจัยผ่านเอ็นเอฟซีบนสมาร์ตโฟน (SECURE HOTSPOTS WITH MULTI-FACTOR AUTHENTICATION THROUGH NFC ON SMARTPHONES) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: รศ. ดร.ญาใจ ลิ่มปิยะกรณ์, 67 หน้า.

ปัจจุบัน เทคโนโลยีเครือข่ายแบบไร้สายเป็นองค์ประกอบสำคัญของธุรกิจ สถานที่ทำงานบางแห่งมีนโยบายให้พนักงานนำอุปกรณ์ที่เป็นของตนเองเพื่อเข้าถึงสารสนเทศและใช้งานแอปพลิเคชันต่างๆของบริษัท หลายองค์กรได้จัดเตรียมระบบเครือข่ายไร้สายภายในองค์กร เพื่อสนับสนุนการเชื่อมต่อให้กับอุปกรณ์ขนาดเล็กและความสะดวกในการจัดการ อย่างไรก็ตาม ความปลอดภัยของเครือข่ายองค์กรจำเป็นต้องมีการจำกัดสิทธิ์ที่เข้มงวด เนื่องจากข้อมูลสำคัญจำนวนมากจะถูกถ่ายโอนผ่านเครือข่ายไร้สาย ดังนั้น การพิสูจน์ตัวจริงถูกมองว่าเป็นหน้าด่านแรกของการป้องกันการเข้าถึงที่ไม่มีสิทธิซึ่งสามารถลดภัยคุกคามต่อเครือข่ายไร้สายได้ WPA2 Enterprise กับมาตรฐาน 802.1X มักถูกนำมาใช้เพื่อจัดการขั้นตอนพิสูจน์ตัวจริงบนเครือข่ายด้วยกรอบงาน EAP โดยเฉพาะกรอบงานประเภท EAP-TLS ที่ใช้ใบรับรองในการพิสูจน์ตัวจริงร่วมกัน ซึ่งให้ความปลอดภัยสูงแต่การใช้งานจริงมีความยุ่งยาก เนื่องจากการจัดการใบรับรองลูกข่าย ด้วยเหตุนี้ การจัดการข้อมูลใบรับรองสำหรับผู้ใช้ทั้งหมดจึงเป็นเรื่องยุ่งยากต่อผู้ดูแลระบบ และผู้ใช้มักจะประสบกับกระบวนการพิสูจน์ตัวจริงที่ซับซ้อน เช่น การติดตั้งใบรับรอง การกำหนดการตั้งค่าเครือข่าย เป็นต้น เพื่อลดภาระในการจัดการใบรับรองของผู้ดูแลระบบ และเพื่อสนับสนุนนโยบายความปลอดภัยเครือข่ายไร้สายระดับสูง งานวิจัยนี้จึงนำเสนอแนวทางการพิสูจน์ตัวจริงกับจุดพร้อมโยงเครือข่ายไร้สายในองค์กรด้วยการใช้เทคโนโลยีเอ็นเอฟซีร่วมกับการพิสูจน์ตัวจริงแบบหลายปัจจัย สำหรับแลกเปลี่ยนข้อมูลใบรับรองและข้อมูลการตั้งค่าเครือข่าย ระบบถูกพัฒนาบนสมาร์ตโฟนแอนดรอยด์ที่สนับสนุนเอ็นเอฟซีโดยใช้งานร่วมกับสถาปัตยกรรมเครือข่ายไร้สายจำลองที่ใช้การตรวจสอบสิทธิ์แบบ WPA2-802.1X กับ EAP-TLS ทั้งนี้แนวทางที่นำเสนอได้ถูกประเมินด้วย ตัววัดเวลาใช้งานจริงในการติดต่อเข้าถึงเครือข่ายไร้สาย และแบบสอบถามการประเมินประสบการณ์การใช้งานจากผู้ใช้

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....

สาขาวิชา วิทยาศาสตร์คอมพิวเตอร์ ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

ปีการศึกษา 2560

5970959621 : MAJOR COMPUTER SCIENCE
KEYWORDS: NEAR FIELD COMMUNICATION, MULTI-FACTOR AUTHENTICATION, WIRELESS NETWORK SECURITY

WIPHOP POMAK: SECURE HOTSPOTS WITH MULTI-FACTOR AUTHENTICATION THROUGH NFC ON SMARTPHONES. ADVISOR: ASSOC. PROF. DR.YACHAI LIMPIYAKORN, 67 pp.

Nowadays, wireless network technology is an essential component of businesses. Some workplaces have adopted the policy bring your own device (BYOD) that permits employees to bring personal devices and to use those devices to access company information and applications. Many organizations provide the wireless local area network (WLAN) to support mobility, connectivity for small equipment and simple management. However, corporate network security requires influence authentication to limit unauthorized access since critical data are transferred over the wireless networks. Hence, an authentication is considered as the first line of preventing unauthorized access that could alleviate Wi-Fi threats. WPA2 Enterprise with 802.1X standard is typically introduced to handle user authentication on the network using the EAP framework. Especially, the EAP-TLS is very secure with allowing for mutual identification but deployments are difficult because of client certificate management. Consequently, credentials management for all users is a hassle for administrators and users usually encounter complicated authentication processes like certificates installation, network configuration setup and so on. To reduce the administrative burden on the certificate management and support network security policies, this research thus presents an approach of Wi-Fi hotspot authentication in the organization by leveraging NFC for credential information exchange in addition to MFA technique adoption. The implementation is developed on NFC-enabled smartphones through network model based on WPA2-802.1X authentication with EAP-TLS. The proposed approach is evaluated by the measure of the actual time of wireless network access, in addition to the questionnaire of the user experience assessment.

Department: Computer Engineering Student's Signature .....
Field of Study: Computer Science Advisor's Signature .....
Academic Year: 2017

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยความอนุเคราะห์อย่างดียิ่งของรองศาสตราจารย์ ดร.ญาใจ ลิ่มปิยะกรณ์ อาจารย์ที่ปรึกษาวิทยานิพนธ์ ซึ่งท่านได้กรุณาสละเวลาให้ความรู้ ให้คำปรึกษา ตรวจสอบ ให้คำแนะนำแนวทางการวิจัย และสนับสนุนจนทำให้การวิจัยในครั้งนี้สำเร็จลุล่วงด้วยดี ข้าพเจ้าจึงขอกราบระลึกถึงพระคุณของรองศาสตราจารย์ ดร.ญาใจ ลิ่มปิยะกรณ์ ไว้ ณ ที่นี้

ข้าพเจ้าขอกราบขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร.สุกรี สินธุภิญโญ และ ดร.ภาสกร อภิรักษ์วรพินิต กรรมการสอบวิทยานิพนธ์ ที่ได้กรุณาสละเวลา ให้คำแนะนำ ตรวจสอบ และแก้ไขวิทยานิพนธ์ฉบับนี้ให้ถูกต้องสมบูรณ์ยิ่งขึ้น

ท้ายนี้ข้าพเจ้าขอกราบขอบพระคุณ คุณพ่อ คุณแม่ และครอบครัวสำหรับกำลังใจที่มีค่ายิ่ง รวมถึงขอขอบพระคุณผู้บังคับบัญชาในสายงาน เพื่อนร่วมงาน และมิตรสหาย ที่คอยติดตามให้กำลังใจ ให้การสนับสนุนและความช่วยเหลือในด้านต่างๆ และท่านอื่นๆ ที่มิได้กล่าวชื่อไว้ ณ ที่นี้ที่มีส่วนช่วยให้วิทยานิพนธ์ของข้าพเจ้าสำเร็จไปได้ด้วยดี

สารบัญ

หน้า

หน้า

หน้า

หน้า

สารบัญตาราง

สารบัญภาพ

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของปัญหา

ในช่วงไม่กี่ปีที่ผ่านมาการใช้งานอินเตอร์เน็ตเพื่อทำธุรกรรมออนไลน์บนอุปกรณ์สมาร์ตโฟนทั้งจากเครือข่ายโทรศัพท์มือถือ และเครือข่ายไร้สายมีจำนวนเพิ่มมากขึ้น ทำให้ความปลอดภัยในการเชื่อมต่อเครือข่ายไร้สายบนมือถือเป็นเรื่องที่ต้องให้ความสำคัญ เนื่องจากการทำธุรกรรมดังกล่าวถูกจัดการผ่าน Wi-Fi ฮอตสปอตมากกว่าเครือข่ายมือถือ ด้วยเหตุผลนี้ทำให้อุปกรณ์โทรศัพท์มีความเสี่ยงต่อการถูกคุกคามความปลอดภัยมากขึ้น โดยเฉพาะอย่างยิ่งในองค์กรที่มีนโยบายนำอุปกรณ์ของคุณมาเอง (BYOD) [1] ที่อนุญาตให้พนักงานนำอุปกรณ์พกพาของตัวเองมาที่ทำงาน และใช้อุปกรณ์ดังกล่าวเพื่อใช้ทรัพยากรที่มีการควบคุมการเข้าถึงของบริษัท เช่น อีเมล ไฟล์เซิร์ฟเวอร์ และฐานข้อมูล เป็นต้น ซึ่งอุปสรรคที่สำคัญคือการติดตาม และควบคุมการเข้าถึงเครือข่ายที่อาจส่งผลกระทบต่อองค์กรภายหลังได้ เช่น ข้อมูลรั่วไหล, ไวรัส, มัลแวร์ และอื่นๆ ดังนั้นการพิสูจน์ตัวจริงจึงถูกให้ความสำคัญเป็นอันดับแรกเพื่อป้องกันเครือข่ายไร้สาย

ในปัจจุบันวิธีการพิสูจน์ตัวจริงแบบเดิมที่ใช้ เช่น ชื่อผู้ใช้, รหัสผ่าน, หรือ Wi-Fi Protected Access (WPA) และ WPA2 นั้นยังปลอดภัยไม่เพียงพอในระดับองค์กร ด้วยเหตุนี้ WPA/WPA2-Enterprise และ RADIUS (802.1X) [2] ที่มาพร้อมกับ Extensible Authentication Protocol (EAP) ถูกสนับสนุนให้มีการนำมาใช้ในองค์กร [3] โดยเฉพาะวิธีการ EAP Transport Layer Security (EAP-TLS) [4] ซึ่งให้ความปลอดภัยสูงสุดสำหรับเครือข่ายไร้สายในปัจจุบัน [5] ข้อดีของ EAP-TLS นั้นใช้ใบรับรองอิเล็กทรอนิกส์ (certificates) เพื่อพิสูจน์ตัวจริงร่วมกันระหว่างลูกข่าย และเซิร์ฟเวอร์ ซึ่งใบรับรองนี้ถูกออกให้โดยหน่วยงานที่เชื่อถือได้ Certificate Authority (CA) อย่างไรก็ตามถึงแม้ว่าวิธีการนี้จะมีความปลอดภัยสูงแต่การใช้งานจริงมีความยากในการจัดการใบรับรองให้กับอุปกรณ์แต่ละเครื่อง ทั้งการแจกจ่าย และติดตั้ง หากขั้นตอนเหล่านี้สามารถกำหนดการตั้งค่าได้อย่างรวดเร็วจะช่วยลดความยุ่งยาก และภาระกับผู้ดูแลระบบ และผู้ใช้ในอุปกรณ์ขนาดเล็กได้ รวมถึงช่วยส่งเสริมนโยบายความปลอดภัยของเครือข่ายในระดับสูงด้วย

งานวิจัยนี้นำเสนอแนวทางการพิสูจน์ตัวจริงกับเครือข่ายไร้สายความปลอดภัยสูงด้วยการใช้เอ็นเอฟซีสนับสนุนในการแลกเปลี่ยนข้อมูลใบรับรอง และประยุกต์ใช้การพิสูจน์ตัวจริงแบบหลายปัจจัย (Multi-Factor Authentication) เพื่อเสริมความความปลอดภัยให้กับเครือข่ายไร้สาย และผู้ใช้งาน ชิ้นงานจะถูกพัฒนาอยู่บนโทรศัพท์สมาร์ตโฟนที่สนับสนุนเอ็นเอฟซีในระบบปฏิบัติการ Android 7.0 ร่วมกับสถาปัตยกรรมเครือข่ายไร้สายที่ใช้ WPA2-Enterprise พร้อมมาตรฐาน IEEE

802.1X กับวิธีการพิสูจน์ตัวจริงแบบ EAP-TLS (certificate-based) ผลลัพธ์ของงานวิจัยที่สนใจคือผลของเวลา และผลการประเมินประสบการณ์ผู้ใช้ ซึ่งผลของเวลาแบ่งออกเป็นสองส่วนได้แก่ 1) เวลาที่ใช้ในการแลกเปลี่ยนข้อมูลใบรับรองอิเล็กทรอนิกส์ และ 2) เวลาที่ใช้ในการพิสูจน์ตัวจริงจนถึงสถานะเชื่อมต่อเครือข่ายสำเร็จ

1.2 วัตถุประสงค์ของงานวิจัย

1.2.1 เพื่อเสนอแนวทางการประยุกต์ใช้งานเอ็นเอฟซีชิฟที่มีอยู่บนโทรศัพท์สมาร์ตโฟนในการสนับสนุนโยบายความปลอดภัยของเครือข่ายไร้สายในระดับองค์กร

1.2.2 เพื่อลดขั้นตอน และความยุ่งยากของการจัดการใบรับรองสำหรับผู้ดูแลเครือข่ายและผู้ใช้งาน เช่น การติดตั้งใบรับรอง, การแจกจ่ายใบรับรอง, การกำหนดการตั้งค่าเครือข่าย เป็นต้น

1.3 ขอบเขตการดำเนินงาน

1.3.1 แบบจำลองเครือข่ายไร้สายที่ถูกสร้างขึ้นอ้างอิงจาก WPA2-Enterprise กับมาตรฐาน IEEE 802.1X ด้วยวิธีการพิสูจน์ตัวจริงแบบ EAP-TLS มีองค์ประกอบดังนี้

- Authentication server (RADIUS server)
- Authenticator (Access Point)
- Supplicant (Smartphone)

1.3.2 ผู้ใช้สามารถรับข้อมูลความลับจากเครื่องบริการ (Authentication server) ผ่านช่องทางเอ็นเอฟซีเทอร์มินอลประกอบด้วย

- ใบรับรองอิเล็กทรอนิกส์ของผู้ใช้ (User.p12)
- ใบรับรองอิเล็กทรอนิกส์ที่ผู้ให้บริการออกให้ หรือ Certificate Authority (CA.crt)
- ข้อมูลการกำหนดการตั้งค่าของเครือข่าย เช่น ชื่อเครือข่าย, ชนิดของ EAP, ชนิดของความปลอดภัย และชื่อผู้ใช้

1.3.3 ผู้ใช้สามารถเชื่อมต่อเครือข่ายไร้สายเป้าหมายอัตโนมัติหลังจากทำการติดตั้งใบรับรองสำเร็จ

1.3.4 การติดตั้งใบรับรองเกิดขึ้นหลังจากผู้ใช้ได้รับข้อมูลใบรับรองจากฝั่งเซิร์ฟเวอร์สำเร็จ

1.3.5 ข้อมูลสำคัญของผู้ใช้บนสมาร์ตโฟนถูกเข้ารหัสด้วยกุญสาธารณะของเซิร์ฟเวอร์ และถูกเก็บไว้ในแฟ้มแอปพลิเคชันด้วยระบบ Keystore

1.3.6 ผู้ดูแลเครือข่ายสามารถจัดการใบรับรองได้ง่ายขึ้น เช่น การสร้างใบรับรองใหม่ การยกเลิกใบรับรองที่ถูกละเมิด หรือการจัดการข้อมูลส่วนผู้ใช้ เป็นต้น

1.3.7 ต้องการแอปพลิเคชันทำงานอยู่ทั้งฝั่งมือถือ และเซิร์ฟเวอร์ซึ่งแอปพลิเคชันถูกควบคุมด้วย Java โปรแกรม

1.3.8 การพิสูจน์ตัวจริงแบบหลายปัจจัย (Multi-Factor Authentication) ถูกนำมาใช้เพื่อเพิ่มความปลอดภัย และความเป็นส่วนตัวให้กับระบบงานมากขึ้น

1.4 ขั้นตอนการวิจัย

1.4.1 ค้นคว้าศึกษา และทำความเข้าใจทฤษฎี และงานวิจัยที่เกี่ยวข้อง

1.4.2 กำหนดขั้นตอน และออกแบบสถาปัตยกรรมเครือข่ายไร้สายโดยอ้างอิงจาก WPA2-Entertrise บนมาตรฐาน 802.1X กับวิธีการพิสูจน์ตัวจริงแบบ EAP-TLS

1.4.3 ทดลองเครือข่ายเพื่อให้สามารถใช้งานได้จริง

1.4.4 กำหนดขั้นตอน และออกแบบแอปพลิเคชันบนสมาร์ตโฟน

1.4.5 ทดลองทำการแลกเปลี่ยนข้อมูลระหว่างสมาร์ตโฟน และเซิร์ฟเวอร์

1.4.6 ปรับแต่ง และพัฒนาฟังก์ชันที่ช่วยให้ผู้ใช้ใช้งานง่าย เช่น ฟังก์ชันการเชื่อมต่อเครือข่ายอัตโนมัติ, ผู้ช่วยติดตั้งใบรับรอง, วิธีการจัดเก็บข้อมูล รวมถึงฟังก์ชันการเข้ารหัส และถอดรหัสข้อมูล

1.4.7 ทดสอบผลลัพธ์จากชิ้นงาน และบันทึกผล

1.4.8 พัฒนา และปรับปรุงโมเดลให้สามารถทำงานได้อย่างถูกต้องตามจุดประสงค์

1.4.9 ทดสอบ และประเมินผลงานวิจัย

1.4.10 ตีพิมพ์ผลงานทางวิชาการ

1.4.11 จัดทำวิทยานิพนธ์

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1.5.1 ได้แนวทางเพื่อลดความยุ่งยากในการกำหนดค่าเครือข่ายบนอุปกรณ์โทรศัพท์สมาร์ตโฟน

1.5.2 ช่วยให้ผู้ดูแลเครือข่ายสามารถจัดการหนังสือรับรองให้กับผู้ใช้ง่ายขึ้น

1.5.3 ช่วยสนับสนุนนโยบายเครือข่ายที่ต้องการความปลอดภัยสูง และใช้การพิสูจน์ตัวจริงด้วยใบรับรองอิเล็กทรอนิกส์

1.5.4 เพิ่มระดับความปลอดภัย และความเป็นส่วนตัวของผู้ใช้จากการประยุกต์ใช้การพิสูจน์ตัวจริงแบบหลายปัจจัย (Multi-Factor Authentication)

1.6 ลำดับการจัดเรียงเนื้อหาในวิทยานิพนธ์

วิทยานิพนธ์นี้มีทั้งหมด 6 บท ดังต่อไปนี้ บทที่ 1 บทนำความเป็นมาและความสำคัญของปัญหาวัถถุประสงค์ของการวิจัย ขอบเขตของการวิจัย ประโยชน์ที่คาดว่าจะได้รับ และผลงานตีพิมพ์ บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง บท 3 แนวคิดและวิธีวิจัย บทที่ 4 การพัฒนาระบบ บทที่ 5 การทดทดสอบและวิเคราะห์ผล บทที่ 6 สรุปผลการวิจัย

1.7 ผลงานที่ตีพิมพ์จากวิทยานิพนธ์

ส่วนหนึ่งของวิทยานิพนธ์ได้รับการตีพิมพ์บทความวิชาการ 2 บทความ ประกอบด้วย

1) W. Pomak and Y. Limpiyakorn, “Enterprise WiFi Hotspot Authentication with Hybrid Encryption on NFC-Enabled Smartphones” ในรายงานการประชุมวิชาการนานาชาติสืบเนื่องจาก The 8th International Conference on Electronics Information and Emergency Communication (ICEIEC 2018), June 15-17, 2018, Beijing, China.
2) W. Pomak and Y. Limpiyakorn, “Secure Hotspot with Multi-Factor Authentication through NFC on Smartphones” ในรายงานการประชุมวิชาการนานาชาติสืบเนื่องจาก The 8th International Workshop on Computer Science and Engineering (WCSE 2018), June 28-30, 2018, Bangkok, Thailand

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 การสื่อสารสนามใกล้ (Near Field Communication) [6]

Near Field Communication (NFC) หรือ การสื่อสารสนามใกล้เป็นเทคโนโลยีการสื่อสารไร้สัมผัสซึ่งอิงตามมาตรฐาน Radio-frequency identification (RFID) ในย่านความถี่สูงช่วยสนับสนุนรองรับการสื่อสารระหว่างอุปกรณ์อิเล็กทรอนิกส์ในระยะใกล้ ๆ เช่น สามาร์ตโฟนหรือแท็บเล็ต เอ็นเอฟซีทำงานที่ความถี่ 13.56 MHz บนอินเตอร์เฟสมาตรฐาน ISO/IEC 18000-3 ซึ่งมีอัตราการถ่ายโอนข้อมูลอยู่ที่ 106, 212 และ 424 Kbits ต่อวินาที เอ็นเอฟซีถูกออกแบบมาให้การแลกเปลี่ยนข้อมูลมีความปลอดภัยด้วยลักษณะการทำงานที่มีระยะเพียงไม่กี่เซนติเมตรระหว่างสองอุปกรณ์ คือประมาณ 20 ซม. ในทางทฤฎี และ ประมาณ 10 ซม. หรือน้อยกว่านั้นในการใช้งานจริงซึ่งสามารถลดโอกาสในการดักฟังข้อมูลได้

2.1.2 การพิสูจน์ตัวจริงแบบหลายปัจจัย (Multi-Factor Authentication) [1]

Multi-Factor Authentication (MFA) หรือ การพิสูจน์ตัวจริงแบบหลายปัจจัย เป็นระบบการรักษาความปลอดภัยที่ต้องการวิธีการพิสูจน์ตัวตนมากกว่าหนึ่งประเภทจากข้อมูลรับรองที่เป็นอิสระต่อกันเพื่อยืนยันตัวตนของผู้ใช้สำหรับการเข้าสู่ระบบหรือการทำธุรกรรมต่างๆ โดยทั่วไปจะมีอย่างน้อย 2 ปัจจัยที่บุคคลสามารถใช้เพื่อพิสูจน์ตัวจริงโดยแบ่งออกเป็น 3 ประเภทดังนี้

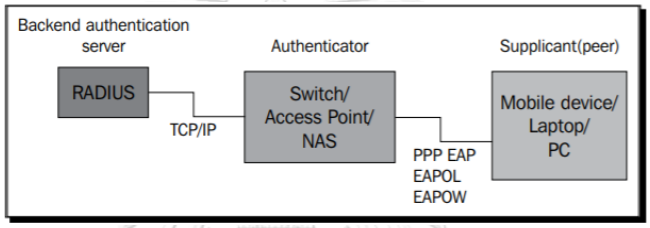
1) Knowledge Factors (Something user know) คือ สิ่งที่ผู้ใช้เท่านั้นที่รู้ เช่น รหัสผ่าน PIN หรือความลับที่ใช้ร่วมกัน (Secret key)

2) Possession Factors (Something user have) คือ สิ่งที่ผู้ใช้มีหรือครอบครองสิ่งนั้น เช่น บัตรประจำตัวประชาชน บัตรประจำตัวความปลอดภัย ATM หรือ Smartphones

3) Inherence Factors (Something user are) หรือ Biometrics คือ สิ่งที่ผู้ใช้เป็นโดยสิ่งเหล่านี้อาจจะเป็นลักษณะเฉพาะส่วนบุคคลหรือลักษณะทางกายภาพของบุคคลนั้น เช่น ลายนิ้วมือ ใบหน้า และเสียง นอกจากนี้ยังรวมไปถึงพฤติกรรมเฉพาะอย่าง เช่น การกดแป้นพิมพ์ การเดินหรือรูปแบบการพูด

2.1.3 IEEE 802.1X [2]

IEEE 802.1X เป็นกรอบมาตรฐาน IEEE สำหรับ port-based network access control หรือมาตรฐานที่ใช้พิสูจน์ตัวจริงผู้ใช้งานที่พยายามเข้าถึงเครือข่ายทั้งแบบมีสาย และแบบไร้สาย ดังแสดงในภาพที่ 1 ซึ่งคือส่วนประกอบหลักสามส่วนของ 802.1X ได้แก่ 1) Supplicant คืออุปกรณ์ client 2) authenticator คือสวิตช์อีเทอร์เน็ตหรือจุดเข้าใช้งานแบบไร้สาย (wireless access point) และ 3) authentication เซิร์ฟเวอร์ คือโฮสต์ที่ใช้งานซอฟต์แวร์ที่สนับสนุนโปรโตคอล RADIUS มาพร้อมกับวิธีการพิสูจน์ตัวจริงแบบต่างๆ จาก EAP เช่น TLS, TTLS, PEAP, MSCHAPV2 เป็นต้น สำหรับ IEEE 802.1X ใช้ EAP เพื่อสร้างอุโมงค์ที่ปลอดภัย (secure tunnel) ระหว่าง client และ authentication เซิร์ฟเวอร์ ในการแลกเปลี่ยนการรับรองความถูกต้อง



ภาพที่ 1 ส่วนประกอบหลักตามมาตรฐาน 802.1X [7]

2.1.4 เทคโนโลยีโครงสร้างพื้นฐานกุญแจสาธารณะ (Public Key Infrastructure)

Public Key Infrastructure (PKI) [8] หรือ เทคโนโลยีโครงสร้างพื้นฐานกุญแจสาธารณะ เป็นใบรับรองอิเล็กทรอนิกส์ (certificate) ที่สามารถพิสูจน์ตัวจริงซึ่งกัน (mutual authentication) และกันได้ระหว่างลูกข่ายกับเซิร์ฟเวอร์ว่าบุคคลอื่นเป็นผู้ที่อ้างสิทธิ์หรือไม่ ใบรับรอง PKI สร้างขึ้นโดยผู้ออกใบรับรอง (Certificate Authority) หรือ CA การออกใบรับรองส่วนตัวจำเป็นต้องติดตั้ง 'Root Certificate' จาก CA ของผู้ให้บริการแต่ละเครื่องก่อนจึงจะสามารถทำให้ เซิร์ฟเวอร์ ยืนยันว่าใบรับรองที่ได้รับมานั้นถูกออกให้จากผู้ให้บริการจริง โดยปกติโครงสร้างพื้นฐานกุญแจสาธารณะจะประกอบไปด้วยสี่องค์ประกอบสำคัญ ได้แก่ 1) ผู้ใช้บริการ (End Entity) เป็นผู้ประสงค์จะขอใช้บริการใบรับรองอิเล็กทรอนิกส์ 2) เจ้าหน้าที่รับลงทะเบียน (Registration Authority) เป็นผู้รับลงทะเบียนจากผู้ร้องขอ 3) ผู้ให้บริการออกใบรับรองอิเล็กทรอนิกส์ (Certification Authority) เป็นองค์กรทำหน้าที่ออกใบรับรองอิเล็กทรอนิกส์ และเพื่อรับรองตัวตน 4) ที่บันทึกข้อมูล (Repository) เป็นระบบคอมพิวเตอร์ที่สามารถสืบค้นใบรับรองอิเล็กทรอนิกส์ของผู้ให้บริการอย่างปลอดภัย

2.1.5 ระบบการเข้ารหัสแบบลูกผสม (Hybrid Cryptosystem)

Hybrid Cryptosystem (HS) หรือ ระบบการเข้ารหัสแบบลูกผสมคือการใช้ข้อดีของการเข้ารหัสแบบสมมาตร (Symmetric encryption) และไม่สมมาตร (Asymmetric encryption มาใช้งานร่วมกัน โดยการเข้ารหัสแบบสมมาตรสนับสนุนความเร็วในการเข้ารหัสข้อมูลเนื่องจากใช้การคำนวณทางคณิตศาสตร์ที่ไม่ซับซ้อน ในขณะที่การเข้ารหัสลับแบบอสมมาตรช่วยให้การสื่อสารมีความปลอดภัยเนื่องจากการเข้ารหัสไม่สมมาตรนั้นไม่จำเป็นต้องใช้คีย์ลับร่วมกัน (shared secret key) ระหว่างผู้ส่ง และผู้รับ โดยปกติแล้วการเข้ารหัสแบบลูกผสมประกอบด้วยสองอัลกอริทึม ตัวอย่างเช่น การใช้วิธีเข้ารหัสแบบ (Advanced Encryption Standard หรือ AES) เป็นวิธีการเข้ารหัสแบบสมมาตรที่สุ่มสร้างคีย์เซสชัน (key session) ที่มีขนาดคีย์ 128, 192 หรือ 256 บิต และบล็อกขนาด 128 บิตซึ่งใช้เพื่อเข้ารหัสข้อความยาว ๆ และการใช้วิธีเข้ารหัสแบบ (Rivest-Shamir-Adleman หรือ RSA) เป็นวิธีการเข้ารหัสแบบไม่สมมาตรที่ใช้ในการเข้ารหัสคีย์เซสชันด้วยคีย์สาธารณะ

2.1.6 การจำลองบัตรด้วยโฮสต์ (Host-Base Card Emulation)

Host-Base Card Emulation (HCE) อนุญาตให้มีสมาร์ตโฟนที่มีเอ็นเอฟซีสามารถเลียนแบบเป็นการ์ด และสื่อสารกับโฮสต์ซีพียูผ่านแอพพลิเคชันที่ทำงานบนแอนดรอยด์ได้โดยตรง ซึ่งไม่จำเป็นต้องใช้หรือพึ่งพา Secure Element (SE) เช่น SIM หรือ ชิป EMV เป็นต้น HCE ถูกออกแบบมาเพื่อลดค่าใช้จ่ายและลดการพึ่งพา SE ซึ่งคุณสมบัตินี้ช่วยให้สามารถแลกเปลี่ยนคำสั่ง APDU (Application Protocol Data Unit) และการตอบสนองในรูปแบบการสื่อสารแบบสองทิศทางผ่านทางเอ็นเอฟซีเทอร์มินัล

2.1.7 EAP Transport Layer Security (EAP-TLS)

EAP-Transport Layer Security (EAP-TLS) ซึ่งกำหนดไว้ใน RFC 5216 เป็นวิธีการรับรองความถูกต้อง IEEE 802.1X EAP ที่ใช้โปรโตคอล TLS (RFC 5246) EAP-TLS สามารถพิสูจน์ตัวจริงร่วมกันระหว่าง supplicant และ authentication เซิร์ฟเวอร์เป็นใบรับรองดิจิทัลแบบ X.509 ซึ่งใช้แนวคิดโครงสร้างคีย์สาธารณะ (PKI) ที่สร้างขึ้นโดยโปรแกรม CA (Certification Authority) ใบรับรองประกอบด้วยชื่อของเซิร์ฟเวอร์หรือผู้ออกซึ่งจำเป็นสำหรับเซิร์ฟเวอร์การตรวจสอบสิทธิ์และสำหรับลูกข่าย EAP-TLS ทุกราย วิธีการพิสูจน์ตัวจริงนี้เป็นวิธีที่ปลอดภัยที่สุดในการใช้งาน WLANs ในปัจจุบัน อย่างไรก็ตามการรักษาใบรับรองของลูกข่ายยังคงเป็นปัญหาสำหรับผู้ดูแลระบบในการจัดการใบรับรอง

2.1.8 Keyed-Hash Message Authentication Code (HMAC)

HMAC ถูกนำมาใช้ตรวจสอบบูรณภาพ (integrity) ของข้อความที่ได้รับ สามารถนำมาใช้ในการส่งข้อมูลผ่านเครือข่ายหรือช่องทางที่ไม่มีความปลอดภัย โดยข้อความนั้นอาจจะถูกเข้ารหัสหรือจะเป็นข้อความเปล่า (plain text) HMAC ช่วยพิสูจน์ได้ว่าข้อความนั้นจะไม่ถูกแก้ไขเปลี่ยนแปลง (alteration) โดยการเปลี่ยนแปลงข้อความเพียงเล็กน้อยสามรถทำให้ค่ารหัสของแฮชมีความแตกต่างกันมาก ซึ่งวิธีนี้คล้ายกับวิธีข้อความย่อย (message digest) เพื่อคำนวณแฮชแต่จะใช้คีย์ลับร่วม (shared secret key) เพื่อให้เฉพาะบุคคลที่มีคีย์ลับร่วมสามารถตรวจสอบความถูกต้องของข้อความ

2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 Secure hotspot authentication through a Near Field Communication side-channel [2]

งานวิจัยนี้นำเสนอเรื่องระบบพิสูจน์ตัวจริงของ Wi-Fi hotspot สาธารณะที่อาศัยเอ็นเอฟซีเพื่อรับข้อมูลการกำหนดการตั้งค่าเครือข่ายและข้อมูลความลับซึ่งถูกเข้ารหัสไว้ล่วงหน้าจากเอ็นเอฟซีแท็กในการพิสูจน์ตัวจริงจะใช้ WPA2 personal หรือ shared-key ซึ่งถูกฝังไว้ในแท็กเช่นกัน นอกจากนี้อุปกรณ์ access point ถูกปรับแต่งให้มีความสามารถในการตรวจสอบได้ว่า ผู้ที่ขอสิทธิ์การเข้าถึงเครือข่ายได้รับข้อมูลความลับมาถูกต้องหรือไม่ ด้วยการถอดรหัสข้อมูลผ่าน secure port

2.2.2 Time-based One-Time Password for Wi-Fi Authentication and Security[9]

การใช้รหัสผ่านแบบใช้ครั้งเดียว (Time-based One-Time Password หรือ TOTP) เพื่อพิสูจน์ตัวจริงกับ Wi-Fi hotspot บนอุปกรณ์อย่างน้อยสองเครื่อง โดยเครื่องหนึ่งทำหน้าที่เป็นจุดเชื่อมต่อ (Hotspot) ในขณะที่อุปกรณ์อื่น ๆ ที่เชื่อมต่อกับจุดเชื่อมต่อนี้ถือเป็น client ข้อมูลความลับ (TOTP code) จะได้รับการสร้างแบบไดนามิกและเปลี่ยนเป็นรหัสการตอบกลับด่วน (Quick Response code: QR code) อย่างไรก็ตาม QR code สามารถถอดรหัสได้ง่ายหากมีตัวอย่างเพียงพอ อัลกอริทึมของการสร้างรหัสผ่านจะสามารถคาดเดาจนสามารถทำ online attack ได้

2.2.3 Email encryption using hybrid cryptosystem based on Android [10]

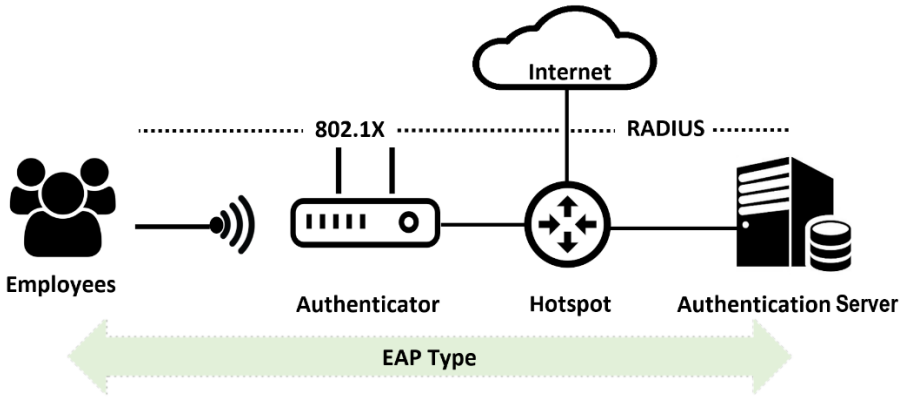
ในงานนี้ใช้เทคนิคระบบการเข้ารหัสแบบลูกผสมในการปิดบังเนื้อหาข้อความในอีเมลบนระบบ Android เพื่อป้องกันข้อมูลสำคัญถูกเปิดเผยและแก้ไขเปลี่ยนแปลง โดยระบบลูกผสมที่ใช้ประกอบไปด้วยสามส่วนหลัก คือ ระบบการเข้ารหัสคีย์สาธารณะ ระบบการเข้ารหัสแบบสมมาตร และฟังก์ชันแฮช (Hash function)

บทที่ 3
แนวคิดและวิธีวิจัย

แนวคิดและวิธีวิจัยของระบบจุดพร้อมโยงที่ปลอดภัยด้วยการพิสูจน์ตัวจริงแบบหลายปัจจัยผ่านเอ็นเอฟซีบนสมาร์ตโฟนสามารถแบ่งออกเป็นทั้งหมด 5 ขั้นตอนหลัก ได้แก่ 1) ออกแบบและสร้างเครือข่ายไร้สายตามมาตรฐาน IEEE 802X กับวิธี EAP-TLS (Network architecture design and implementation) 2) ออกแบบและพัฒนาแอปพลิเคชันทั้งฝั่งสมาร์ตโฟนและฝั่งเซิร์ฟเวอร์ให้สามารถแลกเปลี่ยนข้อมูลกันได้ (Application development) 3) จัดทำฟังก์ชันอำนวยความสะดวกและอัตโนมัติที่จำเป็นเพิ่มเติม (Facilitated and automatic functions) 4) จัดเตรียมและสร้างใบรับรองอิเล็กทรอนิกส์ และ 5) ทำการทดลองใช้งานจริงเพื่อหาค่าเวลาที่ใช้ในส่วนของการแลกเปลี่ยนข้อมูลและการพิสูจน์ตัวจริง (Experimental phase)

3.1 ออกแบบและสร้างเครือข่ายไร้สายตามมาตรฐาน IEEE 802X (EAP-TLS)

ในการสร้างเครือข่ายไร้สายนี้ได้อ้างอิงตามมาตรฐาน IEEE 802.1X โดยใช้ WAP2-Enterprise กับวิธีการพิสูจน์ตัวจริงแบบ EAP-TLS เพื่อเป็นโครงสร้างหลักของงาน ในการให้บริการอินเตอร์เน็ตในระดับองค์กรเสมือนจริงดังแสดงในภาพที่ 2



ภาพที่ 2 ตัวอย่างโครงสร้างเครือข่ายไร้สายตามมาตรฐาน 802.1X กับวิธี EAP-TLS

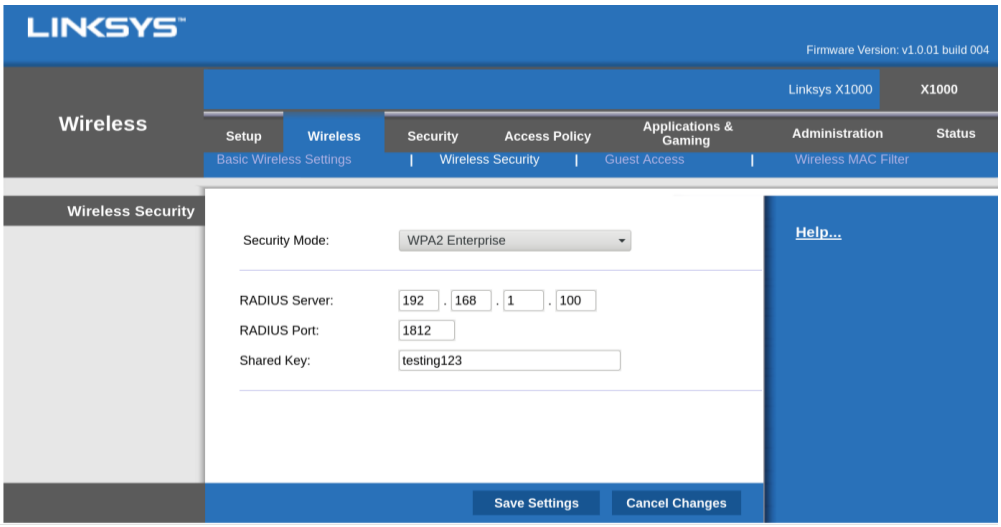
จากภาพที่ 2 โครงสร้างเครือข่ายประกอบไปด้วย 3 ส่วนหลักคือ 1) Authentication เซิร์ฟเวอร์ (RADIUS เซิร์ฟเวอร์) ซึ่งใช้ระบบปฏิบัติการ Centos V.7 2) Supplicant คือบุคคลหรืออุปกรณ์ที่ต้องการเชื่อมต่อ Wi-Fi และ 3) Authenticator (wireless access point) เป็นตัวกลางที่ช่วยในส่งข้อมูลระหว่าง supplicant และ เซิร์ฟเวอร์ซึ่งใช้ Linksys รุ่น X1000

RADIUS server จำเป็นต้องมีการติดตั้งเครื่องมือสำหรับ authenticator ได้แก่ FreeRadius ซอฟต์แวร์ ฐานข้อมูลของ RADIUS Apache service และอื่นๆ ซึ่งจะมาพร้อมกับวิธีการพิสูจน์ตัวจริงหลากหลายชนิดของ EAP ในภาพที่ 3 คือตัวอย่างแฟ้มที่ใช้สำหรับเก็บไฟล์ตั้งค่าต่างๆ ของ FreeRadius

```
/etc/raddb
[root@localhost raddb]# ls -ltr
total 168
-rw-r----- 1 root root     7476 Feb 26 01:45 clients.conf.bak
-rw-r----- 1 root radiusd 28622 Feb 26 22:11 radiusd.conf.bak
-rw-r----- 1 root radiusd  7581 Feb 27 01:43 clients.conf
drwxr-xr-x 2 root radiusd     6 Mar 20 04:57 test
drwxrwx--- 2 root radiusd  4096 Apr  7 12:37 certs.bak
drwxr-xr-x 2 root root       57 Apr 23 23:26 backup
-rw-r----- 1 root radiusd  8536 May 16 13:33 trigger.conf
-rw-r----- 1 root radiusd  3470 May 16 13:33 templates.conf
-rw-r----- 1 root radiusd 20808 May 16 13:33 README.rst
-rw-r----- 1 root radiusd 28621 May 16 13:33 radiusd.conf
-rw-r----- 1 root radiusd 28361 May 16 13:33 proxy.conf
-rw-r----- 1 root radiusd    52 May 16 13:33 panic.gdb
-rw-r--r-- 1 root radiusd  1440 May 16 13:33 dictionary
lrwxrwxrwx 1 root radiusd    35 May 22 22:47 huntgroups -> ./m
lrwxrwxrwx 1 root radiusd    30 May 22 22:47 hints -> ./mods-c
drwxrwx--- 2 root radiusd  4096 May 22 22:47 certs
drwxr-x--- 6 root radiusd    85 May 22 22:47 mods-config
drwxr-x--- 2 root radiusd  4096 May 22 22:47 mods-available
drwxr-x--- 2 root radiusd  4096 May 22 22:47 mods-enabled
drwxr-x--- 2 root radiusd   145 May 22 22:47 policy.d
drwxr-x--- 2 root radiusd    41 May 22 22:47 sites-enabled
drwxr-x--- 2 root radiusd  4096 May 22 22:47 sites-available
lrwxrwxrwx 1 root radiusd    29 May 22 22:47 users -> ./mods-c
```

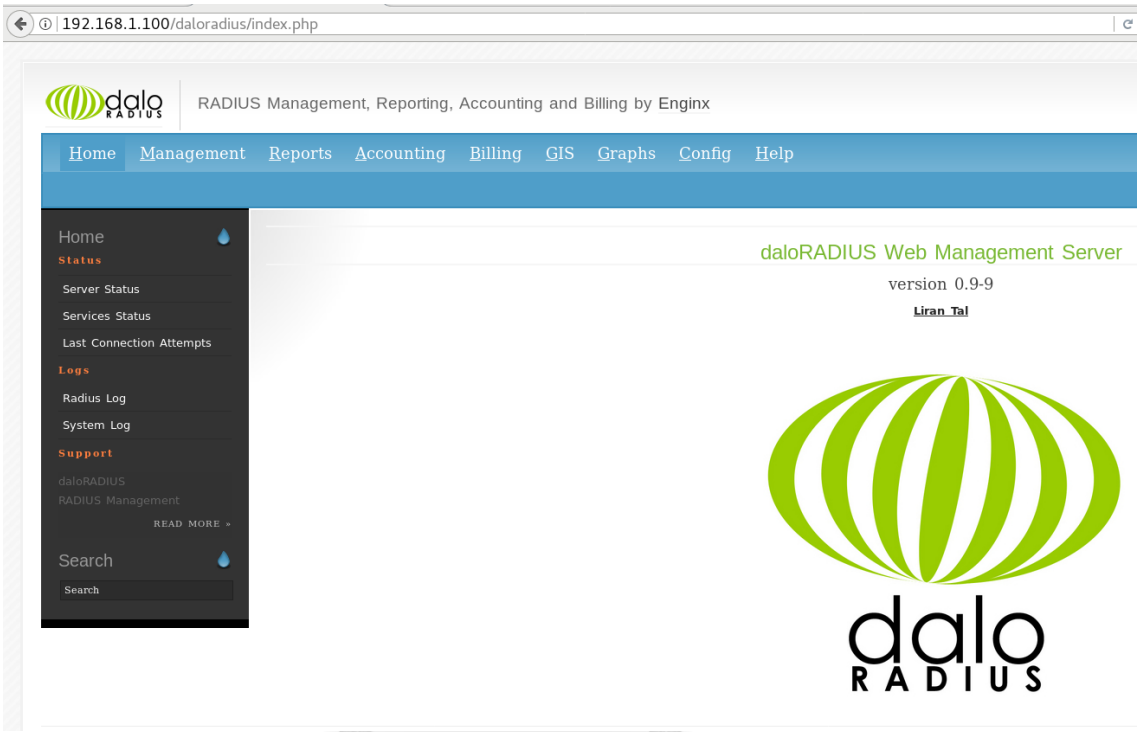
ภาพที่ 3 ตัวอย่างแฟ้มที่ใช้สำหรับเก็บไฟล์ตั้งค่าต่างๆ ของ FreeRadius

การตั้งค่าให้กับ authenticator หรือ wireless access point โดยกำหนดให้ใช้ WPA2-Enterprise ใส่ IP address ของ Authentication server, shared secret key และ port มาตรฐานเพื่อให้สามารถเชื่อมต่อไปยัง RADIUS เซิร์ฟเวอร์ ดังแสดงในภาพที่ 4



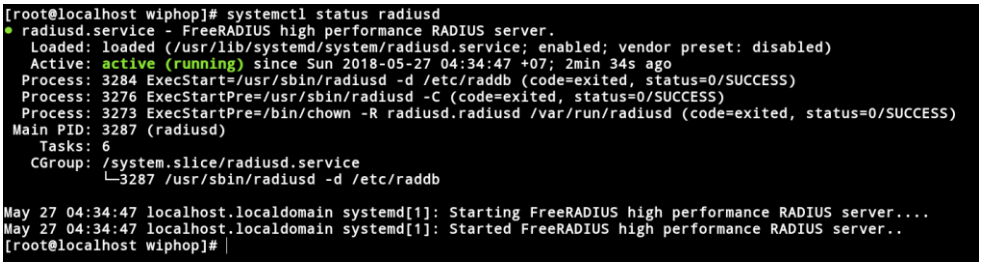
ภาพที่ 4 กำหนดการตั้งค่าให้กับ wireless access point

หลังจากที่ทำการติดตั้งระบบและตั้งค่าระบบทุกอย่างเรียบร้อยแล้ว ขั้นตอนต่อไปคือทดสอบการใช้งานโดยเข้าไปยังหน้าเว็บของเครื่องมือ FreeRadius ผ่านเว็บเบราว์เซอร์โดยใส่รายละเอียดดังนี้คือ192.168.1.100/daloradius/index.php ซึ่ง 192.168.1.100 นี้คือ IP ประจำตัวของ RADIUS server หากหน้าเว็บสามารถแสดงผลดังภาพที่ 5 จึงถือว่า RADIUS server นั้นสามารถทำงานได้อย่างถูกต้อง



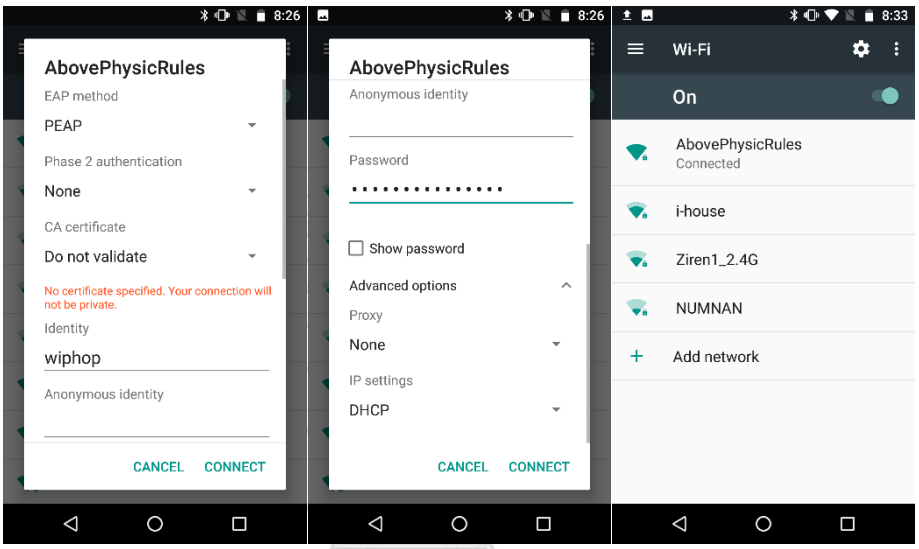
ภาพที่ 5 การแสดงผล FreeRadius ที่ทำงานได้อย่างถูกต้อง

นอกจากนี้ยังต้องมีการทดสอบสถานะของบริการ RADIUS ผ่าน command line “systemctl status radiusd” หากพบว่าสถานะคือ Active ถือว่าเครื่องมือสามารถทำงานได้ปกติ ดังภาพที่ 6



ภาพที่ 6 สถานะ Active แสดงถึงบริการ Radius ทำงานได้อย่างถูกต้อง

ขั้นตอนสุดท้ายคือทดสอบการเชื่อมต่อกับโครงสร้างเครือข่ายดังภาพที่ 7 ซึ่งประกอบไปด้วย 3 ภาพ โดยภาพที่ 7 (ก) และภาพที่ 7 (ข) นั้นเป็นส่วนที่ผู้ใช้พยายามทำการเชื่อมต่อ Wi-Fi ชื่อ “AbovePhysicRules” ตามที่กำหนดตั้งค่าไว้คือ WAP2-Enterprise คือต้องมีการกรอกชื่อผู้ใช้คือ “wiphop” และ รหัสผ่าน นอกจากนี้ยังต้องทำการกำหนด EAP method ด้วยซึ่งในที่นี้ใช้วิธี EAP-PEAP สำหรับทดสอบ โดยจะเปลี่ยนเป็น EAP-TLS ภายหลัง และภาพที่ 7 (ค) คือสถานะการเชื่อมต่อ Wi-Fi เป็น connected หรือ เชื่อมต่อแล้ว หากการพิสูจน์ตัวจริงสำเร็จ

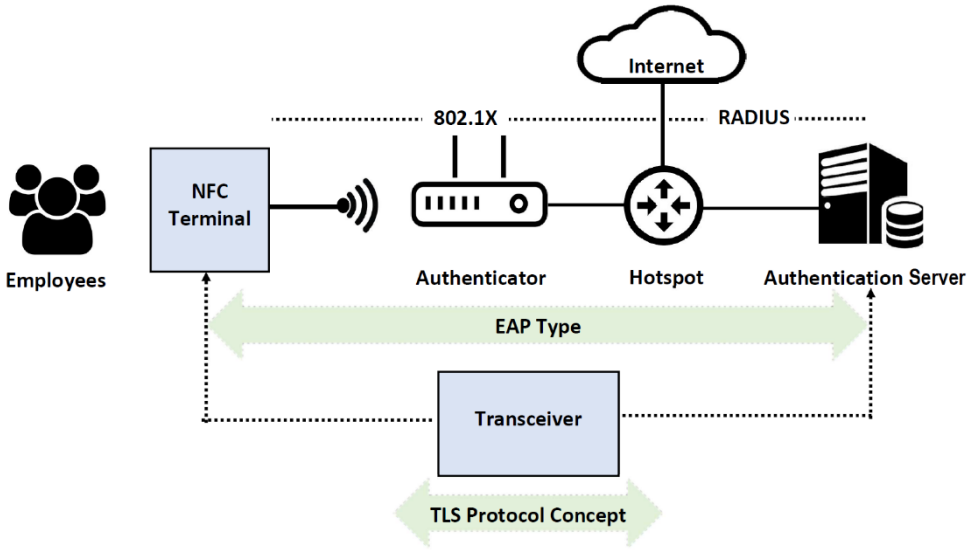


(ก) (ข) (ค)

ภาพที่ 7 รายละเอียดและสถานะการเชื่อมต่อ Wi-Fi ผ่านเครือข่าย WPA2-Enterprise

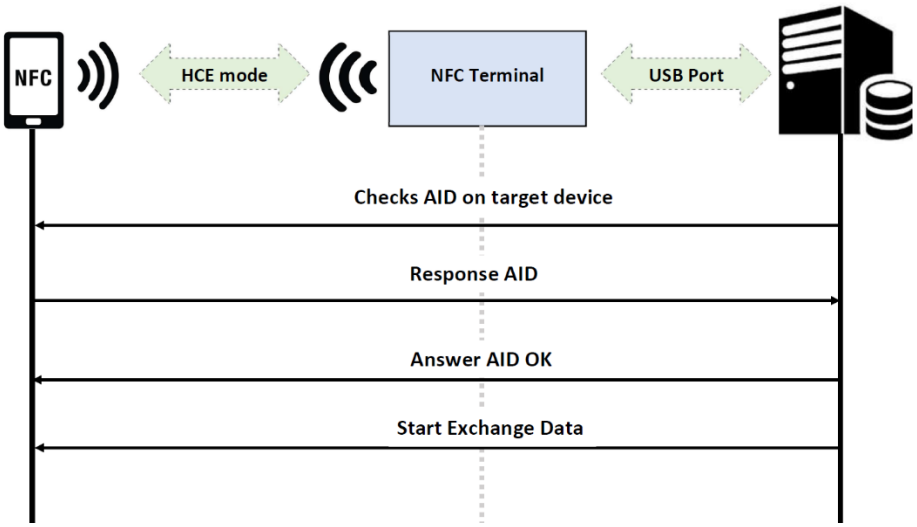
3.2 ออกแบบและพัฒนาแอปพลิเคชันทั้งฝั่งสมาร์ตโฟนและฝั่งเซิร์ฟเวอร์ให้สามารถแลกเปลี่ยนข้อมูลกันได้

การออกแบบแอปพลิเคชันโครงสร้างเครือข่ายไร้สายจากขั้นตอน 3.1 จะเชื่อมต่อกับตัวอ่านเอ็นเอฟซี (ACR122U) ซึ่งทำหน้าที่เป็นตัวรับและส่งข้อมูล (transceiver) ให้กับสมาร์ตโฟนและเครื่องบริการดังแสดงในภาพที่ 8



ภาพที่ 8 ตัวรับและส่งข้อมูลเชื่อมต่อกับเครือข่ายไร้สายหลัก

สำหรับการทำงานของตัวรับและส่งข้อมูลจะทำหน้าที่ตรวจสอบเลขประจำตัวแอปพลิเคชัน (Application ID หรือ AID) เครื่องสมาร์ตโฟนเป้าหมายทุกครั้งก่อนที่จะทำการแลกเปลี่ยนข้อมูล ซึ่งในขั้นตอนนี้ถือเป็นขั้นตอนเริ่มต้น (initiate stage) ดังแสดงในภาพที่ 9



ภาพที่ 9 สถานะเริ่มต้นก่อนที่จะทำการแลกเปลี่ยนข้อมูล

3.3 จัดทำฟังก์ชันอำนวยความสะดวกและอัตโนมัติที่จำเป็นเพิ่มเติม

ในการจัดทำฟังก์ชันอำนวยความสะดวกและอัตโนมัติที่จำเป็นเพิ่มเติมให้กับแอปพลิเคชันถูกพัฒนาหลังจากสถาปัตยกรรมเครือข่ายไร้สายที่กำหนดสามารถเชื่อมต่อได้และแอปพลิเคชันหลัก

สามารถใช้งานได้จริง จากนั้นจะสร้างฟังก์เพิ่มเติม เช่น ฟังก์ชันการเข้ารหัสและถอดรหัส ฟังก์ชันจัดการส่งไฟล์ใบรับรอง ฟังก์ชันตรวจสอบข้อมูลผู้ใช้กับฐานข้อมูล ฟังก์ผู้ช่วยติดตั้งใบรับรอง และฟังก์ชันเชื่อมต่อเครือข่ายให้อัตโนมัติทั้งฝั่งสมาร์ตโฟนและฝั่งเครื่องบริการ

3.4 จัดเตรียมและสร้างใบรับรองอิเล็กทรอนิกส์ให้กับผู้ใช้

การพิสูจน์ตัวจริงจำเป็นต้องสร้างใบรับรองอิเล็กทรอนิกส์เพื่อใช้ตรวจสอบสิทธิ์ผู้ใช้ก่อนการเชื่อมต่อเครือข่ายไร้สาย ซึ่งใบรับรองอิเล็กทรอนิกส์ที่ต้องการจะประกอบไปด้วยสองใบ ได้แก่ 1) ใบรับรองผู้ใช้ (user certificate) และ 2) ใบรับรองของผู้ออกใบรับรองหลัก (root certificate authority หรือ root CA) โดยใบรับรองผู้ใช้จำเป็นต้องใช้เพื่อตรวจสอบสิทธิ์ลูกข่าย และใบรับรอง root CA เพื่อใช้ยืนยันว่าใบรับรองได้ถูกออกให้โดยองค์กรหรือเจ้าหน้าที่จริง

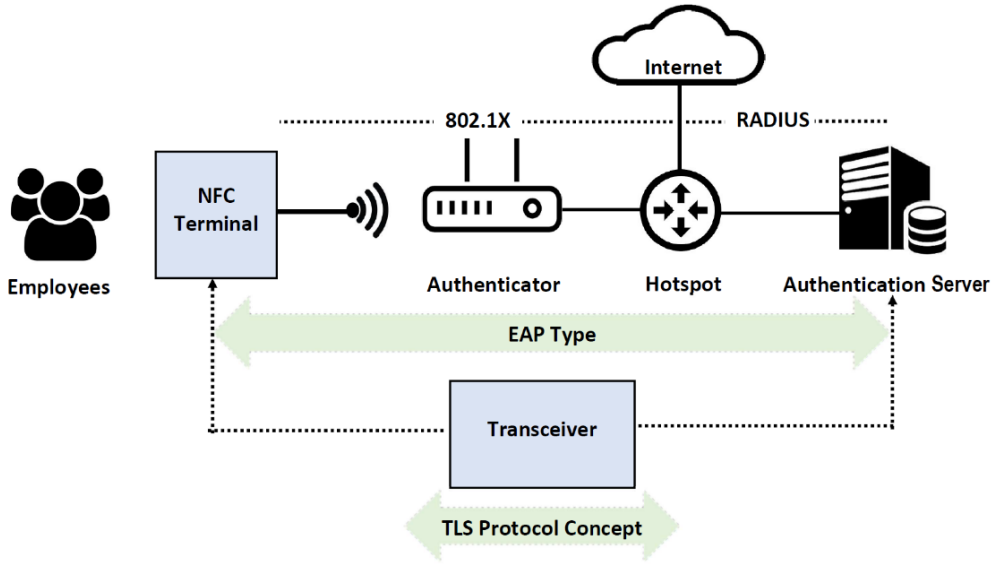
3.5 ทำการทดลองใช้งานจริงเพื่อหาค่าเวลาที่ใช้ในส่วนของการแลกเปลี่ยนข้อมูลและการพิสูจน์ตัวจริง

เพื่อเป็นการทดสอบการใช้งานจึงต้องมีการนำชิ้นงานไปทดลองและเก็บค่าเวลาเพื่อหาค่าเฉลี่ยความมั่นใจของสองส่วนหลัก คือ 1) เวลาที่ใช้ในการรับข้อมูลความลับจากฝั่ง เซิร์ฟเวอร์ และ 2) เวลาที่ใช้ในการพิสูจน์ตัวจริงจนถึงสถานะเชื่อมต่อเครือข่ายสำเร็จ นอกจากนี้ผลลัพธ์ของงานวิจัยที่สนใจคือผลการประเมินประสบการณ์ผู้ใช้

บทที่ 4
การพัฒนาระบบ

4.1 สถาปัตยกรรมระบบ

สถาปัตยกรรมของระบบสามารถแบ่งออกเป็น 6 ส่วนหลัก ได้แก่ 1) ตัวรับและส่งข้อมูล (Transceiver) 2) ส่วนการสร้างใบรับรองจากฝั่งเครื่องบริการ (Certificates provision) 3) ตัวจัดการใบรับรอง (Certificate manager) 4) ตัวจัดเก็บข้อมูลหนังสือรับรอง (Credential storage) 5) ตัวจัดการเครือข่าย (Network manager) และ 6) ฟังก์ชันที่จำเป็นในแอปพลิเคชัน (Functions) โดยภาพรวมการทำงานของระบบแสดงดังภาพที่ 10 ซึ่งมีรายละเอียดดังนี้



ภาพที่ 10 ภาพรวมการทำงานของระบบ

4.2 สภาพแวดล้อมและเครื่องมือที่ใช้ในการพัฒนา

สภาพแวดล้อม และเครื่องมือที่ใช้ในการพัฒนาระบบ ประกอบด้วยรายการฮาร์ดแวร์ และซอฟต์แวร์ดังต่อไปนี้

4.2.1 สภาพแวดล้อม

สภาพแวดล้อมสามารถแบ่งออกเป็น 5 ส่วนดังนี้

1. วินโดวส์ 10

1.1. หน่วยประมวลผลอินเทล คอร์ ไอ 7-7500U 2.70 กิกะเฮิร์ต (CPU Intel Core i7-7500U 2.7GHz)

1.2. หน่วยความจำ 8 กิกะไบต์ (8 GB RAM)

1.3. ฮาร์ดดิสก์ความจุ 1 เทระไบต์ (1 TB HDD)

1.4. ระบบปฏิบัติการไมโครซอฟต์วินโดวส์ 10 (Microsoft Windows 10) แบบ 64 บิท

2. Android smartphone

2.1. Motorola G5 Plus

2.2. microSD, up to 256 GB

2.3. ระบบปฏิบัติการ Android 7.0 (Nougat), upgradable to Android 8.0 (Oreo)

2.4. Chipset Qualcomm MSM8953 Snapdragon 625

2.5. CPU Octa-core 2.0 GHz Cortex-A53

3. CentOS Version 7.0

3.1. หน่วยประมวลผลอินเทล คอร์ ไอ 7-7500U 2.70 กิกะเฮิร์ต (CPU Intel Core i7-7500U 2.7GHz)

3.2. หน่วยความจำ 4 กิกะไบต์ (4 GB RAM)

3.3. ฮาร์ดดิสก์ความจุ 30 กิกะไบต์ (30 GB HDD)

3.4. ระบบปฏิบัติการลินุกซ์ แบบ 64 บิท

4. NFC Reader

4.1 ACR122U USB NFC Reader - Advanced Card Systems Ltd

5. Wireless Access Point

5.1 LINKSYS รุ่น X1000

4.2.2 เครื่องมือที่ใช้ในการพัฒนา

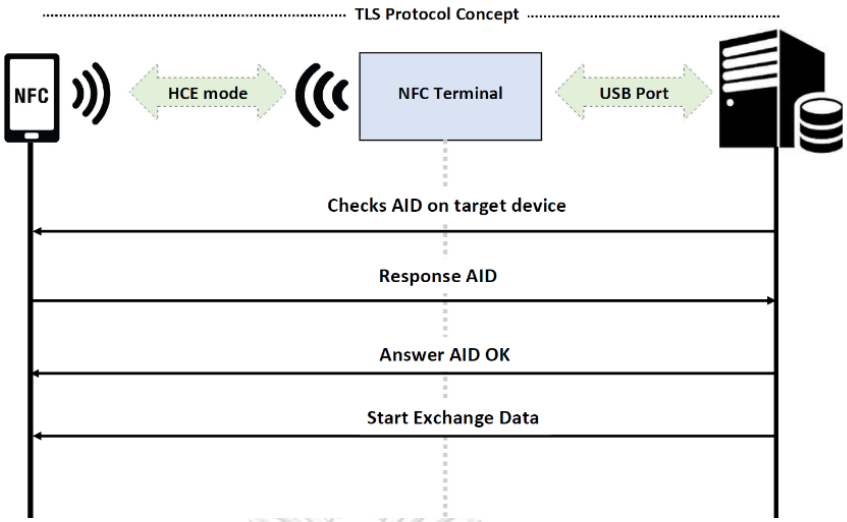
1. แอนดรอยด์สตูดิโอ เวอร์ชัน 3.1.2 (Android Studio 3.1.2)
2. อีคิปส์ เวอร์ชัน 4.7 (Eclipse 4.7)
3. Openssl tool
4. Wireshark

4.3 วิธีและขั้นตอนการพัฒนา

4.3.1 ตัวรับและส่งข้อมูล (Transceiver)

ตัวรับและส่งข้อมูลเป็นส่วนสำคัญของระบบงานนี้ เนื่องจากใช้เพื่อจัดส่งใบรับรอง รับและส่งคำสั่ง ขนส่งข้อมูลของเครือข่าย และข้อมูลอื่นๆ ซึ่งจำเป็นต้องสร้างแอปพลิเคชันทั้งบนสมาร์ตโฟนและเอ็นเอฟซีเดสก์ท็อป (NFC Desktop) ให้สามารถแลกเปลี่ยนข้อมูลกันได้ โดยการสื่อสารที่เกิดขึ้นจะพึ่งพาบริการ Host-based Card Emulation (HCE) และคำสั่ง Application Data Protocol Unit (APDUs) ซึ่งเป็นการสื่อสารแบบสองทิศทาง (bidirectional communication) ในการส่งคำสั่งและรับการตอบสนองคำสั่งซึ่งจะอยู่ในรูปแบบของเลขฐานสิบหก

การพัฒนาแอปพลิเคชันบนมือถือจะใช้โปแกรมแอนดรอยด์สตูดิโอ เวอร์ชัน 3.1.2 เพื่อสร้างฟังก์ชันพื้นฐานหลักที่สำคัญในการสื่อสารระหว่างสมาร์ตโฟนและเอ็นเอฟซีเดสก์ท็อป ซึ่งบริการ HCE (HCE service) บนสมาร์ตโฟนสนับสุนการแลกเปลี่ยนข้อมูลผ่านเอ็นเอฟซีระหว่างมือถือและตัวอ่านเอ็นเอฟซีซึ่งภาพที่ 11 แสดงถึงขั้นตอนเริ่มต้นก่อนการแลกเปลี่ยนข้อมูลระหว่างสมาร์ตโฟนกับฝั่งเครื่องบริการผ่านเอ็นเอฟซีโดยใช้ HCE ที่สามารถเรียกใช้ด้วยการสืบทอดจากคลาสหลัก HostApduService ดังแสดงในภาพที่ 12 และกำหนดสอง abstract method ในคลาสได้แก่ processCommandApdu ซึ่งใช้เพื่อรับส่งข้อมูลคำสั่งด้วย APDUs ดังภาพที่ 13 และ onDeactivated ใช้เพื่อล้างค่าและคืนค่าสถานะเริ่มต้นเมื่อผู้ใช้ยกเลิกการแตะอุปกรณ์กับตัวอ่านเอ็นเอฟซี กรณีการเชื่อมต่อไม่สมบูรณ์หรือการร้องขอ AID ที่ต่างบริการกัน ดังภาพที่ 14



ภาพที่ 11 กระบวนการเริ่มต้นก่อนการแลกเปลี่ยนข้อมูล

```
package com.wiphop.mynfc;

import ...

public class MyHostApduService extends HostApduService{
    int i = 0;
    byte[] temApdu = null;
    int count = 1;
    String text = null;
    boolean mCheck = false;
```

ภาพที่ 12 คลาส MyHostApduService สืบทอดมาจาก คลาสหลัก HostApduService

```
@Override
public byte[] processCommandApdu(byte[] apdu, Bundle extras)
{
    String s = new String( apdu );
    Log.d( tag: "HCE", msg: "APDU To Hex Code : " + byteToHex( apdu ) );
    Log.d( tag: "HCE", msg: "APDU Plaintext : " + s );
        //Check AID
        if (selectAidApdu( apdu ))
            {
                //tStart = System.currentTimeMillis();
                Log.d( tag: "HCE", msg: "Found AID & Application selected " + i++ );
                return DictInf.HAIDSelectionOkid;
            }
```

ภาพที่ 13 Abstract processCommandApdu

```
@Override
public void onDeactivated(int reason){
        try {
            Log.d( tag: "HCE", msg: "Deactivated: " + reason );
            Thread.sleep( millis: 1000 );
            if(mCheck == true) {
                Intent mInten = new Intent( action: "android.nfc.cardemulation.action.NOTIFY_HCE_DATA" );
                sendBroadcast( mInten );
            }
        }catch (InterruptedException e){
            e.printStackTrace();
        }
    }
```

ภาพที่ 14 Abstract onDecactivated

เมื่อผู้ใช้แตะสมาร์ตโฟนที่ตัวอ่านเอ็นเอฟซี ระบบแอนดรอยด์จะต้องทราบว่าบริการ HCE ตัวใดที่ต้องการจะสื่อสารด้วย จึงมีข้อกำหนดตามมาตรฐาน ISO/IEC 7816-4 เพื่อกำหนดวิธีการเลือกแอปพลิชันจาก AID ซึ่ง AID นี้จะประกอบไปด้วย 16 ไบต์ที่กำหนดไว้ล่วงหน้าคือ “F0010203040506” นอกจากนี้จะต้องระบุกลุ่มของ AID โดยปกติจะมีสองกลุ่มหลักคือ CATEGORY_OTHER และ CATEGORY_PAYMENT สำหรับงานนี้กำหนดให้ AID อยู่ในกลุ่มของ CATEGORY_OTHER ซึ่งข้อมูลทั้งหมดถูกกำหนดไว้ในไฟล์ apduservice.xml ดังภาพที่ 15

```
<?xml version="1.0" encoding="utf-8"?>
<host-apdu-service xmlns:android="http://schemas.android.com/apk/res/android"
    android:description="servicedesc"
    android:requireDeviceUnlock="false">
    <aid-group android:description="NFC test"
        android:category="other">
        <aid-filter android:name="F0010203040506"/>
    </aid-group>
</host-apdu-service>
```

ภาพที่ 15 AID ของแอปพลิเคชันประกาศไว้ล่วงหน้าในกลุ่ม CATEGORY_OTHER

การแลกเปลี่ยนข้อมูลผ่านเอ็นเอฟซีในโหมด HCE เริ่มต้นด้วยการใช้คำสั่ง APDU "SELECT AID" ซึ่งถูกส่งจากโปรแกรมฝั่งเซิร์ฟเวอร์ไปยังสมาร์ตโฟนเป้าหมายเพื่อใช้ระบุถึง AID ผ่านบริการ HCE เพื่อตรวจสอบและเริ่มแลกเปลี่ยนข้อมูลผ่านทางเมท็อด processCommandApdu(byte[], Bundle) ซึ่งกระบวนการแลกเปลี่ยนข้อมูลจะเสร็จสิ้นหรือยกเลิกก็ต่อเมื่อ การเชื่อมต่อเอ็นเอฟซีไม่สมบูรณ์ และบริการ HCE ได้รับคำสั่ง APDU "SELECT AID" ที่อ้างอิงถึงบริการแอปพลิชันอื่นในเวลาเดียวกัน

ภาพที่ 16 แสดงถึงตัวอย่างการประกาศเมท็อด createSelectAidApdu เพื่อสร้างคำสั่ง APDU “SELECT AID” ของโปรแกรมทางฝั่งเซิร์ฟเวอร์

```
//8. This method is used to create AID
private byte[] createSelectAidApdu(byte[] aidAndroid)
    {
        byte[] result = new byte[6 + aidAndroid.length];
        System.arraycopy(CLA_INS_P1_P2, 0, result, 0, CLA_INS_P1_P2.length);
        result[4] = (byte)aidAndroid.length;
        System.arraycopy(aidAndroid, 0, result, 5, aidAndroid.length);
        result[result.length - 1] = 0;
    return result;
    }
```

ภาพที่ 16 ประกาศเมท็อดเพื่อสร้างคำสั่ง APDU “SELECT AID”

สำหรับรายละเอียดคำสั่ง APDU ดังแสดงในตารางที่ 1 ซึ่งคำสั่ง APDU ประกอบไปด้วยสองส่วนหลัก ได้แก่ 1) คำสั่งส่วนหัว (Header) มีขนาด 4 ไบต์ และ 2) ข้อมูลที่จะถูกขนส่ง (Payload) ซึ่งตัวอ่านเอ็นเอฟซีโมเดล ACR122u มีความสามารถในการขนส่งข้อมูลมากสุดที่ 256 ไบต์ต่อครั้ง โดยมีรายละเอียดและความหมายดังนี้

ความหมายของคำสั่ง (Command) คือ SELECT_AID

- CLS หรือ (Class) คือ การระบุชนิดของคำสั่ง เช่น ชนิดคำสั่งตามข้อตกลงของผู้ผลิตแต่ละราย (มีขนาด 1 ไบต์)
- INS หรือ (Instruction) คือ รหัสคำสั่งซึ่งใช้เพื่อระบุคำสั่งเฉพาะ เช่น เขียน อ่าน หรือเลือก ในที่นี้ 0XA4 คือรหัสคำสั่ง SELECT (มีขนาด 1 ไบต์)
- P1-P2 หรือ (Parameter 1 – Parameter 2) คือ พารามิเตอร์สำหรับคำสั่ง เช่น ตำแหน่ง offset ที่จะเขียนข้อมูลลงในไฟล์ (มีขนาด 2 ไบต์)
- LE หรือ (length) คือ ความยาวของของข้อมูลที่ถูกขนส่งในที่นี้คือ ความยาวของ AID นั่นคือ “F0010203040506” (มีขนาด 0, 1, 2 หรือ 3 ไบต์)

ตารางที่ 1 โครงสร้างหลักส่วนหัว (Header) และ ส่วนข้อมูล (Payload) ของคำสั่ง APDU

Command	Class	INS	P1	P2	Le	Payload
SELECT_AID	0X00	0XA4	0x04	0x00	0x07	{0X00.,..}

ก่อนการส่งคำสั่ง APDUs จากฝั่งเซิร์ฟเวอร์ผ่านทางตัวอ่านเอ็นเอฟซีไปยังสมาร์ตโฟนนั้น จำเป็นต้องมีการตรวจสอบสถานะของตัวอ่านเอ็นเอฟซีว่าพร้อมใช้งานหรือไม่ ซึ่งตัวอ่านเอ็นเอฟซีจะถูกควบคุมการทำงานด้วยโปรแกรมเอ็นเอฟซีเดสก์ท็อปจากฝั่งเซิร์ฟเวอร์ด้วยการใช้แพ็กเกจ Smart Card I/O API ซึ่งเป็น Java API สำหรับใช้สื่อสารกับ Smart Cards โดยการใช้คำสั่ง APDU ของมาตรฐาน ISO/IEC 7816-4 ที่กล่าวไปข้างต้น โดยตัวอย่างการประกาศใช้ Smart Card API ดังแสดงในภาพที่ 17

```
private CardTerminal getTerminal() throws CardException
    {
        //Terminal Factory <available terminals>
        TerminalFactory factory = TerminalFactory.getDefault();
        List<CardTerminal> terminals = factory.terminals().list();
        System.out.println("Terminals: " + terminals);
        System.out.print("\n");

        //Use First Terminal
        CardTerminal terminal = terminals.get(0);
        return terminal;
    }
```

ภาพที่ 17 ประกาศเมท็อดเพื่อเปิดใช้ Smart Card API

จากนั้นประกาศเมท็อดสำหรับตรวจสอบสถานะของตัวอ่านเอ็นเอฟซีว่าพร้อมใช้งานใช้งานหรือไม่ หากพบสถานะตัวอ่านพร้อมใช้งานแล้ว จะแสดงผลลัพธ์ออกทางหน้าจอเพื่อแสดงถึงสถานะตัวอ่านพร้อมใช้งานและรอส่งคำสั่ง “SELECT AID” ไปยังเครื่องเป้าหมาย ดังภาพที่ 18

```
private void waitTillDevicePresent() throws CardException, InterruptedException
    {
        while(!terminal.isCardPresent()) //Check if device available
            {
                System.out.print("\nNo card detected on device !");
                Thread.sleep(1000); // 1 second
            }
    }
```

ภาพที่ 18 ประกาศเมท็อดเพื่อตรวจสอบสถานะตัวอ่านเอ็นเอฟซี

หากสถานะตัวอ่านเอ็นเอฟซีพร้อมใช้งาน เมท็อด estabblishConnection จะทำการกำหนดค่าให้กับตัวอ่านเอ็นเอฟซีได้แก่ communication protocol และ channel mode ดังภาพที่ 19

```
private void establishConnection() throws InterruptedException, IOException,
                                    GeneralSecurityException, CardException
    {
        //private final byte[] AID_ANDROID = { (byte)0xF0, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06 };
        selectAid = createSelectAidApdu(AID_ANDROID);  //Android Application ID: F0010203040506

        boolean connectionOk = false;
        while(!connectionOk)
            {
                try
                    {
                        deviceAsCard = terminal.connect("DIRECT"); //Reader communication protocols
                        channel = deviceAsCard.getBasicChannel();
                        connectionOk = true;
                    }
                catch (Exception e2)
                    {
                        System.out.println("WAITING FOR CONNECTION");
                        transferDone();
                    }
            }
    }
```

ภาพที่ 19 เมท็อดกำหนดค่าให้กับตัวอ่านเอ็นเอฟซี

โดยภาพที่ 20 แสดงตัวอย่างสถานะตัวอ่านเอ็นเอฟซีที่พร้อมใช้งานซึ่งรอส่งคำสั่ง APDU “SELECT AID ไปยังสมาร์ตโฟนเป้าหมายเพื่อแลกเปลี่ยนข้อมูล

```
Terminals: [PC/SC terminal ACS ACR122U 00 00]

Wait Till Device Present........

No card detected on device !
No card detected on device !
No card detected on device !
```

ภาพที่ 20 สถานะตัวอ่านเอ็นเอฟซีที่พร้อมใช้งาน

เมื่อประกาศสร้างคำสั่ง APDU “SELECT AID”และตรวจสอบสถานะตัวอ่านเอ็นเอฟซีสำเร็จแล้ว เมท็อด selectAppByAID จะทำการส่งคำสั่ง SELECT AIDไปยังสมาร์ตโฟนเพื่อเริ่มต้นแลกเปลี่ยนข้อมูล ดังแสดงในภาพที่ 21

```
  private void selectAppByAID() throws CardException, InterruptedException,
                                IOException, GeneralSecurityException
      {
          System.out.println("\n");
          System.out.print("input: ");
          printByteArray(selectAid);
          boolean AidSelectionSuccess = false;
          String s = "";
          readerResponse = null;

           while(!AidSelectionSuccess)
              {
                 try
                     {
                         //send APDU command to select AID
                         readerResponse = channel.transmit(new CommandAPDU(selectAid));
                                 System.out.print("\nSelection successful. Connection established!\n"
                                         + "Response after AID Selection: ");

                                 //Convert byte to string
                                 s = new String(readerResponse.getBytes(), "US-ASCII");
                                 System.out.println(s);
```

```
                    //If successful, the program continues
                    if(Arrays.equals(DictInf.HAIDSelectionOkid, readerResponse.getBytes()))
                        {
                                try
                                    {
                                        communicate();
                                    }
                                catch (CardException | InterruptedException e)
                                    {
                                        e.printStackTrace();
                                    }
                        }
                    else
                        {
                            AidSelectionSuccess = false;
                            System.out.print("\nNo AID exists, no successful connection !");
                            deviceAsCard.disconnect(true); //Terminate connection
                        }
                }
            catch(Exception e)
                {
                System.out.println("\nError...Communication is reconnecting: ");
                reConnect();
                }
        }
}
```

ภาพที่ 21 ประกาศเมท็อดเพื่อใช้ส่งคำสั่ง APDU “SELECT AID”

เมื่อสมาร์โฟนได้รับคำสั่ง บริการ HCE จะตรวจสอบ AID เฉพาะที่ลงทะเบียนไว้ หากพบจะทำการเปิดแอปพลิเคชันดังกล่าวและตอบกลับด้วยคำสั่ง “AIDSelectionOk” ไปยังโปรแกรมฝั่งเซิร์ฟเวอร์ ดังแสดงในภาพที่ 22 ซึ่งหากฝั่งเซิร์ฟเวอร์ได้รับคำสั่งตอบกลับ “AIDSelectionOk” จะเริ่มกระบวนการแลกเปลี่ยนข้อมูล โดยภาพที่ 23 คือตัวอย่างการส่งข้อมูลและผลลัพธ์ที่ได้หากตรวจสอบ AID สำเร็จ

```
@Override
public byte[] processCommandApdu(byte[] apdu, Bundle extras)
{
    String s = new String( apdu );
    Log.d( tag: "HCE", msg: "APDU To Hex Code : " + byteToHex( apdu ) );
    Log.d( tag: "HCE", msg: "APDU Plaintext : " + s );
        //Check AID
        if (selectAidApdu( apdu ))
            {
                //tStart = System.currentTimeMillis();
                Log.d( tag: "HCE", msg: "Found AID & Application selected " + i++ );
                return DictInf.HAIDSelectionOkid;
            }
```

ภาพที่ 22 ตรวจสอบ AID และตอบกลับด้วย AIDSelectionOk

```
Terminals: [PC/SC terminal ACS ACR122U 00 00]

Wait Till Device Present........

No card detected on device !
No card detected on device !
No card detected on device !

input: 0x00 0xA4 0x04 0x00 0x07 0xF0 0x01 0x02 0x03 0x04 0x05 0x06 0x00
Selection successful. Connection established!
Response after AID Selection: AIDSelectionOk
```

ภาพที่ 23 ตัวอย่างผลลัพธ์เมื่อพบ AID เป้าหมาย

4.3.2 การสร้างใบรับรองจากฝั่งเครื่องบริการให้กับลูกข่าย (Certificate creation)

การสร้างใบรับรองบนฝั่งเครื่องบริการจำเป็นต้องใช้เครื่งมือคำสั่ง openssl อย่างไรก็ตามหลังที่ติดตั้งซอฟต์แวร์ FreeRadius บนระบบปฏิบัติการ Centos7 เสร็จแล้วนั้น แฟ้มกำหนดการตั้งค่าได้จัดเตรียมคำสั่งหรือสคริปต์ (script) เพื่ออำนวยความสะดวกให้กับผู้ดูแลระบบ ดังนั้นขั้นตอนการสร้างใบรับรองนี้จะแก้ไขไฟล์กำหนดการที่ตั้งค่าไว้ล่วงหน้าตามความต้องการของผู้ดูแลระบบ โดยกระบวนการทั้งหมดประกอบไปด้วยสามส่วนหลักดังนี้

4.3.2.1 สร้างใบรับรอง Root Certificate Authority (Root CA)

ในงานนี้ใบรับรอง root CA นั้นจำเป็นต้องใช้เพื่อพิสูจน์ว่าใบรับรองผู้ใช้นั้นถูกให้โดยเจ้าหน้าที่ ซึ่งผู้วิจัยเปรียบเสมือนผู้ดูแลเครือข่าย ดังนั้นผู้วิจัยคือผู้ออกใบรับรองด้วยตนเองจุดประสงค์เพื่อใช้แสดงตัวตนและพิสูจน์ว่าเป็นใบรับรองตัวจริงจากผู้ดูแลเครือข่าย โดยขั้นตอนการสร้างมีดังนี้

1) แก้ไขและอัพเดทไฟล์ ca.cnf ด้วยคำสั่ง “vi ca.cnf” โดยแก้ไขฟิลด์ "input_password" และ "output_password" เป็นรหัสผ่านสำหรับใบรับรอง CA นอกจากนี้รายละเอียดของผู้ออกใบรับรองจำเป็นต้องใส่ เช่น ประเทศ เมือง ชื่อองค์กร อีเมล และชื่อใบรับรอง เป็นต้น ดังภาพที่ 24

```
[ req ]
prompt                  = no
distinguished_name      = certificate_authority
default_bits            = 2048
input_password          = whatever
output_password         = ;b4r1604!
x509_extensions         = v3_ca

[certificate_authority]
countryName             = TH
stateOrProvinceName     = Bangkok
localityName            = Ihouse
organizationName        = Secure WiFi Inc.
emailAddress            = admin@ihouse.org
commonName              = "Secure WiFi Certificate Authority"

[v3_ca]
subjectKeyIdentifier    = hash
authorityKeyIdentifier  = keyid:always,issuer:always
basicConstraints        = critical,CA:true
crlDistributionPoints   = URI:http://www.ihouse.org/ihouse_ca.crl
```

ภาพที่ 24 แก้ไขและใส่ข้อมูลของผู้ออกใบรับรอง

2) ขั้นตอนถัดไปสร้างใบรับรองด้วยคำสั่ง “make ca.pem” โดยเนื้อหาของไฟล์ใบรับรอง ca.pem จะแสดงดังภาพที่ 25

```
-----BEGIN CERTIFICATE-----
MIIE8TCCA9mgAwIBAgIJAOv56jR4iGSXMA0GCSqGSIb3DQEBCwUAMIGYMQswCQYD
VQQGEwJUSDEQMA4GA1UECAwHQmFuZ2tvazEPMA0GA1UEBwwGSWhvdXNlMRkwFwYD
VQQKDBBTZWN1cmUgV2lGaSBJbmMuMR8wHQYJKoZIhvcNAQkBFhBhZG1pbkBpaG91
c2Uub3JnMSowKAYDVQQDDCFTZWN1cmUgV2lGaSBDZXJ0aWZpY2F0ZSBBdXRob3Jp
dHkwHhcNMTgwNjI1MTkyODI2WhcNMjgwNTAzMTkyODI2WjCBmDELMAkGA1UEBhMC
VEgxEDAOBgNVBAgMB0Jhbmdrb2sxDzANBgNVBAcMBktlob3VzZTEZMBcGA1UECgwQ
U2VjdXJlIFdpRmkgSW5jLjEfMB0GCSqGSIb3DQEJARYQYWRtaW5AaWhvdXNlLm9y
ZzEqMCgGA1UEAwwhU2VjdXJlIFdpRmkgQ2VydGlmaWNhdGUgQXVOaG9yaXR5MIIB
IjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA4enDb7sh9ZsuTlexfZgtDs5J
KQfEWth+OYUJ3ZtjizQjMwUqQflZqr31qu29Bjt2kRef5yppIAf+/Qy97soXLxDh
q4ZFkNCvRyVn68q2sAfJ+J5CQWRnVRdniUuUU4JNqkSWytRiSZCp8YGIKMdBqpAF
49/TTpV3O9a5z1+Bfl5DQ7wKMC6etZTxIE4hs7KElo2Yf0C+6WpxlL2WfcJ6WQDf
/3sEuXqEVNMICX6kj5SjODo9Dt7lcmnickxQ7/hqKOpYwprApu8jwntlu7S1mSYE
g1ByeFdx9yVt5FNIOOkJD7ypenISnxGfwCrDP5R8Ds8zLDWEbd6Eg0xtWgUJowID
AQABo4IBOjCCATYwHQYDVR0OBBYEFCSjgVVhoDotct0+XBEjW1gkjqqnMIHNBgNV
HSMEgcUwgcKAFCSjgVVhoDotct0+XBEjW1gkjqqnoYGepIGbMIGYMQswCQYDVQQG
EwJUSDEQMA4GA1UECAwHQmFuZ2tvazEPMA0GA1UEBwwGSWhvdXNlMRkwFwYDVQQK
DBBTZWN1cmUgV2lGaSBJbmMuMR8wHQYJKoZIhvcNAQkBFhBhZG1pbkBpaG91c2Uu
b3JnMSowKAYDVQQDDCFTZWN1cmUgV2lGaSBDZXJ0aWZpY2F0ZSBBdXRob3JpdHmC
CQDr+eoOeIhklzAPBgNVHRMBAf8EBTADAQH/MDQGA1UdHwQtMCswKaAnoCWGI2h0
dHA6Ly93d3cuaWhvdXNlLm9yZy9paG91c2VfY2EuY3JsMA0GCSqGSIb3DQEBCwUA
A4IBAQBe+VumlzJNh+OKo5iwZ54GGqQTQFVCji7kavkGtoV1Ibe5KSt74QA9byBF
U3cCN4eavKcdvxHHoMc3dATvPVkdUYMY5+uOmn+3PSpAVi3S9DW91wgFd5lnL34W
Q9OeOQNOy/DcSjev/c272TFgPpUvdGzNyTxCyVEiqHnaHwVh4ThUfSykbg7Wrhji
/dGDGzRfOAdIuQYO6WxBpZjTT0mq4LDmgOTlbWYfojNb4uPwGP1LhjHne5ef7a/d
tDG26X9iNUB8hXGUG7RUCodwAEJOythhJ2druH1AWOUNMJUYTBxTZFvyD+dNrFXX
y8teijaKAofmwKpubYu8SzAPnDrr
-----END CERTIFICATE-----
```

ภาพที่ 25 เนื้อหาของไฟล์ใบรับรอง ca.pem

4.3.2.2 สร้างใบรับรองของเครื่องบริการ (server certificate)

ใบรับรับรองของเครื่องบริการถูกสร้างขึ้นเพื่อใช้ตรวจสอบใบรับรองที่แจกจ่ายให้ไปยังผู้ใช้แต่ละราย ดังนั้นรหัสผ่านจำเป็นสำหรับใบรับรองเซิร์ฟเวอร์ในฟิลด์ "input_password" และ "output_password" ของไฟล์ server.cnf

1) แก้ไขและปรับไฟล์ server.cnf ด้วยคำสั่ง “vi server.cnf” โดยแก้ไขฟิลด์ "input_password" และ "output_password" เป็นรหัสผ่านสำหรับใบรับรองเซิร์ฟเวอร์ จากนั้น

แก้ไขส่วน [server] เพื่อให้ได้ค่าที่ถูกต้องสำหรับประเทศ จังหวัด ชื่อองค์กร เป็นต้น ซึ่งต้องตรวจสอบให้แน่ใจว่าฟิลด์ commonName นั้นแตกต่างจาก commonName กับใบรับรอง CA ดังภาพที่ 26

```
[ policy_anything ]
countryName             = optional
stateOrProvinceName     = optional
localityName            = optional
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

[ req ]
prompt                  = no
distinguished_name      = server
default_bits            = 2048
input_password          = PassWOrd!
output_password         = PassWOrd!

[server]
countryName             = TH
stateOrProvinceName     = Bangkok
localityName            = Ihouse
organizationName        = Secure WiFi Inc.
emailAddress            = admin@ihouse.org
commonName              = "Secure WiFi Server Certificate"
```

ภาพที่ 26 แก้ไขและใส่ข้อมูลของผู้ออกใบรับรอง

2) ขั้นตอนถัดไปสร้างใบรับรองด้วยคำสั่ง “make server.pem” โดยเนื้อหาของไฟล์ใบรับรอง ca.pem จะแสดงดังภาพที่ 27

```
Bag Attributes
    localKeyID: 55 F2 E8 9F 8F 36 CD 64 53 F5 AF B9 03 AB FE 15 3A E8 C3 C7
subject=/C=TH/ST=Bangkok/O=Secure WiFi Inc./CN=Secure WiFi Server Certificate/emailAddress=admin@ihouse.org
issuer=/C=TH/ST=Bangkok/L=Ihouse/O=Secure WiFi Inc./emailAddress=admin@ihouse.org/CN=Secure WiFi Certificate Autho
-----BEGIN CERTIFICATE-----
MIID6DCCAtCgAwIBAgIBATANBgkqhkiG9w0BAQsFADCBmDELMAkGA1UEBhMCVEgx
EDAOBgNVBAgMB0Jhbmdrb2sxDzANBgNVBAcMBklob3VzZTEZMBcGA1UECgwQU2Vj
dXJlIFdpRmkgSW5jLjEfMB0GCSqGSIb3DQEJARYQYWRtaW5AaWhvdXNlLm9yZzEq
MCgGA1UEAwwhU2VjdXJlIFdpRmkgQ2VydGlmaWNhdGUgQXV0aG9yaXR5MB4XDTE4
MDYyNTE5MjgyNloXDTI4MDUwMzE5MjgyNlowgYQxCzAJBgNVBAYTAlRIMRAwDgYD
VQQIDAdCYW5na29rMRkwFwYDVQQKDBBTZWN1cmUgV2lGaSBJbmMuMScwJQYDVQQD
DB5TZWN1cmUgV2lGaSBTZXJ2ZXIgQ2VydGlmaWNhdGUxHzAdBgkqhkiG9w0BCQEW
EGFkbWluQGlob3VzZS5vcmcwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIB
AQDCCAsjA4u/PBqKv/7qd1pMYHjiAZyPtbbwgpi6er8U6KhSqWTXkW85Zf6EVio9
LTZ72cZ/BhgV+hm7YJilEpxcKlujiwKLsJBuQ61u1yQq5CRf6+xPlCln1jmqijKZ
jmpv05l1zlr4dYjfqSQOVetUj4E2wdChJh1iKvTb4apvYIUOy2rP51umF8aDxd4V
rx+EeVVhD7JhUsPM93nu/z7AOI+8sgtl/DQx+mcgXQL5G1uoHi+fFva4hpppw6as
8ISv1A2KuevTiBf84KwyaRaUWhhO7GqaXF2AbayAFV8VAdNy2mLsbYTnlpPMzGJD
9jej8OCjwhGYk6hlYtolHOxVAgMBAAGjTzBNMBMGA1UdJQQMMAoGCCsGAQUFBwMB
MDYGA1UdHwQvMC0wK6ApoCeGJWh0dHA6Ly93d3cuZXhhbXBsZS5jb20vZXhhbXBs
ZV9jYS5jcmwwDQYJKoZIhvcNAQELBQADggEBAG/sn7EVTRm76AA6157yETUv4oU/
4KO4NID/90Te72BkbY5PccqdJ2MnTvnyV2xO9limkXgrhR+czoY4I5sv2+ar6nYo
48GN8p7A7wUd8inl8iZbc4cOg4YBtu8BAm+6oe2UvE/ejJhS8j/gjycPVhObVCyd
1NOzJgjAcb4y0hpECSa2vl7nOT68FOn6YBmPJrObPR9DVIKxyHFOXRF3PPvRoMmh
WvxJjdP/YUe3J5zekY7LJhsUuWPp3IIawoYQSJcOVq71LYXYZHSiyXUfHoRIk1oe
LBzaWTDL3fSSz7gUrsw61s6aoaRFqo9L1bxXWXO2/1qYc9ofoWHRg7sE2/E=
-----END CERTIFICATE-----
```

ภาพที่ 27 เนื้อหาของไฟล์ใบรับรอง server.pem

4.3.2.3 สร้างใบรับรองผู้ใช้ (user certificate)

ใบรับรองผู้ใช้จะถูกใช้โดยกรอบงาน EAP-TLS EAP-TTLS และ PEAP ซึ่งวิธีการสร้างใบรับรองผู้ใช้นี้จะถูกที่ลงชื่อโดยใบรับรองเครื่องบริการที่สร้างขึ้นด้านในขั้นที่ 4.3.1.2

1) แก้ไขและอัพเดทไฟล์ client.cnf ด้วยคำสั่ง “vi client.cnf” โดยแก้ไขฟิลด์ "input_password" และ "output_password" เป็นรหัสผ่านสำหรับใบรับรองผู้ใช้ รหัสผ่านจำเป็นสำหรับขั้นตอนการติดตั้งใบรับรองแก่ผู้ใช้ปลายทางที่จะใช้ใบรับรอง

2) ขั้นตอนต่อมาคือ แก้ไขส่วน [client] เพื่อให้มีค่าที่ถูกต้อง เช่น ประเทศ จังหวัด ชื่อองค์กร เป็นต้น และฟิลด์ commonName ซึ่งส่วนนี้คือชื่อผู้ใช้ที่จะใช้สำหรับขั้นการพิสูจน์ตัวจริง ดังภาพที่ 28

```
[ req ]
prompt                  = no
distinguished_name      = client
default_bits            = 2048
input_password          = ;b4r1604!
output_password         = ;b4r1604!

[client]
countryName             = TH
stateOrProvinceName     = Bangkok
localityName            = Ihouse
organizationName        = Secure WiFi Inc.
emailAddress            = john@example.org
commonName              = john@example.org
```

ภาพที่ 28 แก้ไขและใส่ข้อมูลของผู้ออกใบรับรอง

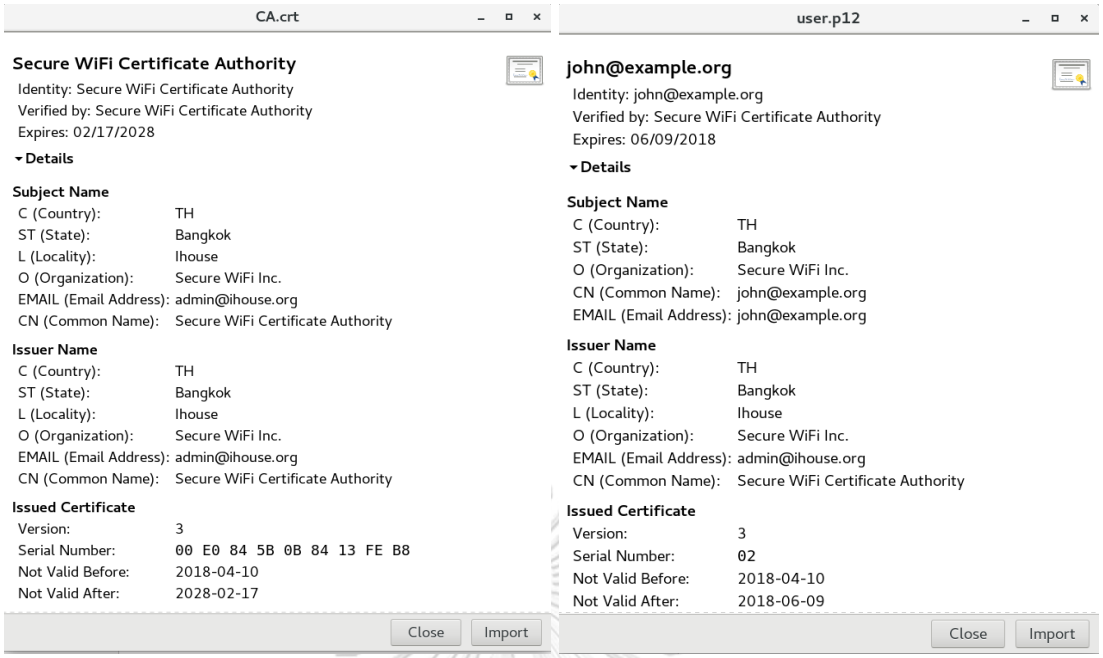
3) สุดท้ายใช้คำสั่ง “make client.pem” ผลลัพธ์คือ ใบรับรองผู้ใช้จะอยู่ในรูปแบบ "emailAddress.pem" เช่น "user@example.com.pem" ในการสร้างใบรับรองสำหรับผู้ใช้รายอื่นสามารถทำซ้ำในขั้นตอน 1) ถึง 3) โดยระบุชื่อในส่วนของ "commonName" และรหัสผ่าน แตกต่างตามแบบฟอร์มที่ผู้ใช้ได้ร้องขอใบรับรอง ดังแสดงในภาพที่ 29

```
Bag Attributes
    localKeyID: 88 4E 42 3F 07 BF B8 EE 5C 5D FC 24 BA 25 55 4D 59 8F 71 C1
subject=/C=TH/ST=Bangkok/O=Secure WiFi Inc./CN=john@example.org/emailAddress=john@example.org
issuer=/C=TH/ST=Bangkok/L=Ihouse/O=Secure WiFi Inc./emailAddress=admin@ihouse.org/CN=Secure WiFi Certificate Author
-----BEGIN CERTIFICATE-----
MIID2TCCAsGgAwIBAgIBAzANBgkqhkiG9w0BAQsFADCBmDELMAkGA1UEBhMCVEgx
EDAOBgNVBAgMB0Jhbmdrb2sxDzANBgNVBAcMBklob3VzZTEZMBcGA1UECgwQU2Vj
dXJlIFdpRmkgSW5jLjEfMB0GCSqGSIb3DQEJARYQYWRtaW5AaWhvdXNlLm9yZzEq
MCgGA1UEAwwhU2VjdXJlIFdpRmkgQ2VydGlmaWNhdGUgQXVOaG9yaXR5MB4XDTE4
MDYyNTIyNTE1NloXDTE4MDgyNDIyNTE1NlowdjELMAkGA1UEBhMCVEgxEDAOBgNV
BAgMB0Jhbmdrb2sxGTAXBgNVBAoMEFNlY3VyZSBXaUZpIEluYy4xGTAXBgNVBAMM
EGpvaG5AZXhhbXBsZS5vcmcxHzAdBgkqhkiG9w0BCQEWEGpvaG5AZXhhbXBsZS5v
cmcwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDiV1sS/41zbobCd8La
H5eK1rWeZ5jVzPR97HJ3C4c1/1pHlYNwBduqtvAGnv+gdfE9lp9Uixx1mtEEOkNV
fD+NaaOOzxWdjRbPQ9aVFaF9iCrW5WBC6Mh8IoHnxfJ7llXdIEvMvpmIiR9Vtp5u
rPBBc1+FlbqF42sI3tIBUyrx+JKYX/2MDtwHlt0fTBAje8LjijrnPpfzpdSLYP4X
zfVFtpiY2zrtCGRtTg6sXQL4IMeQVG2ZE8NEuxUVVBXqW/18T5w5olT4VMkLx4Dp
EeaN9ampuxEAfipjNHifu3snxxeGJtSgDHjv/OaPjBmCARLYC8qxeFn5tZcrvoLW
ON31AgMBAAGjTzBNMBMGA1UdJQQMMAoGCCsGAQUFBwMCMDYGA1UdHwQvMC0wK6Ap
oCeGJWh0dHA6Ly93d3cuZXhhbXBsZS5jb20vZXhhbXBsZV9jYS5jcmwwDQYJKoZI
hvcNAQELBQADggEBAJEirEgV9YiySbbT9Y34UlWFPle3jXOZi1fW2Lw8/Zna2B2d
+lvkaJmO9NC6frPIrdZX4qksTlHzgaBOmfAJhEFmjKqTGtOudp3brXmdKElVo7Qt
Xz5PKdIO6PmWOiOSlqIqq37wbcWMxnEz+oFAnra4+pEVbr8tQNanlAu5h3iOe+P+
x68IghCNfKU59OFGNR2ygx1XUd78WjtbnouXYENDnmqCaI7LNKUjCH3KayrEghtT
3qibazMGEJut0LY+sAiBHzYktYpZXuCEnnlJzPFbxFbFx++jHjGkkbCOjOD+Fjh0
IfXEj+F+dWNd1YEH1xpOOE1zANTvFL7E33OlmvK=
-----END CERTIFICATE-----
```

ภาพที่ 29 เนื้อหาของไฟล์ใบรับรองผู้ใช้ user.pem

4.3.3 ตัวจัดการใบรับรอง (Certificate manager)

ตัวจัดการใบรับรองทำหน้าที่สนับสนุนการติดตั้งใบรับรองที่ได้รับจากผู้ให้บริการ ซึ่งใบรับรองที่จำเป็นต้องใช้ในการติดตั้งประกอบด้วย 2 ส่วน คือ ใบรับรองที่ออกให้โดยเจ้าหน้าที่ (CA.crt) และใบรับรองตัวผู้ใช้ (user.p12) โดยมีรายละเอียดใบรับรอง ดังภาพที่ 30



(ก) (ข)

ภาพที่ 30 ใบรับรองออกโดยเจ้าหน้าที่ (ก) และใบรับรองตัวผู้ใช้ (ข)

ฟังก์ชันผู้ช่วยติดตั้งใบรับรองที่ออกให้โดยเจ้าหน้าที่ (CA.crt) ดังแสดงในดังภาพที่ 31

```
public void installPkcs12Cert() {
    try {
            stop_pkc = true;
            InputStream is = new FileInputStream(CLIENT_CERT);
            byte[] keychain = new byte[is.available()];
            is.read( keychain );
            Intent installIntent = KeyChain.createInstallIntent();
            installIntent.putExtra( KeyChain.EXTRA_PKCS12, keychain);
            installIntent.putExtra( KeyChain.EXTRA_NAME, value: "Client.p12" );
            startActivityForResult( installIntent, requestCode: 0 );
            onActivityResult( requestCode: 1,INSTALL_PKCCERTIFICATE_REQUEST,installIntent);
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

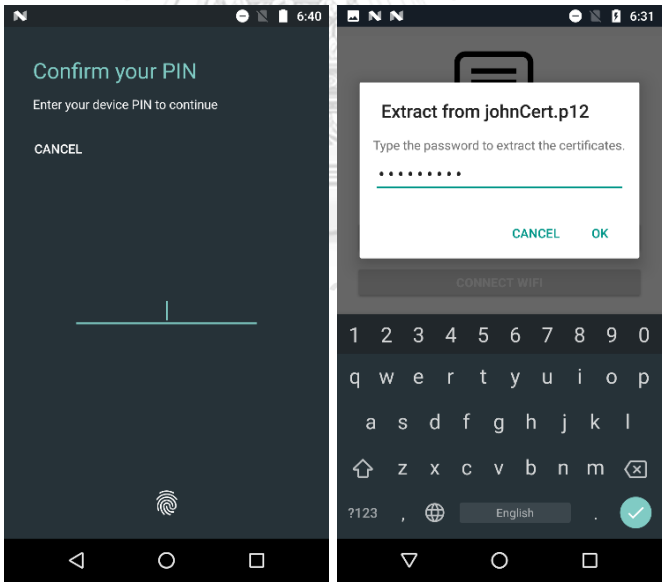
ภาพที่ 31 ฟังก์ชันผู้ช่วยในการติดตั้งใบรับรองที่ออกให้โดยเจ้าหน้าที่ (CA.crt)

ฟังก์ชันผู้ช่วยติดตั้งใบรับรองตัวผู้ใช้ (user.p12) ดังแสดงในดังภาพที่ 32

```
public void installCaCert(){
    try {
            stop_ca = true;
            InputStream is = new FileInputStream(CA_CERT);
            byte[] keychain = new byte[is.available()];
            is.read( keychain );
            Intent installIntent = KeyChain.createInstallIntent();
            installIntent.putExtra( KeyChain.EXTRA_CERTIFICATE, keychain );
            installIntent.putExtra( KeyChain.EXTRA_NAME,  value: "CA.crt" );
            startActivityForResult( installIntent, INSTALL_CACERTIFICATE_REQUEST );
            onActivityResult( requestCode: 0,INSTALL_CACERTIFICATE_REQUEST,installIntent);
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

ภาพที่ 32 ฟังก์ชันผู้ช่วยในการติดตั้งใบรับรองตัวผู้ใช้ (user.p12)

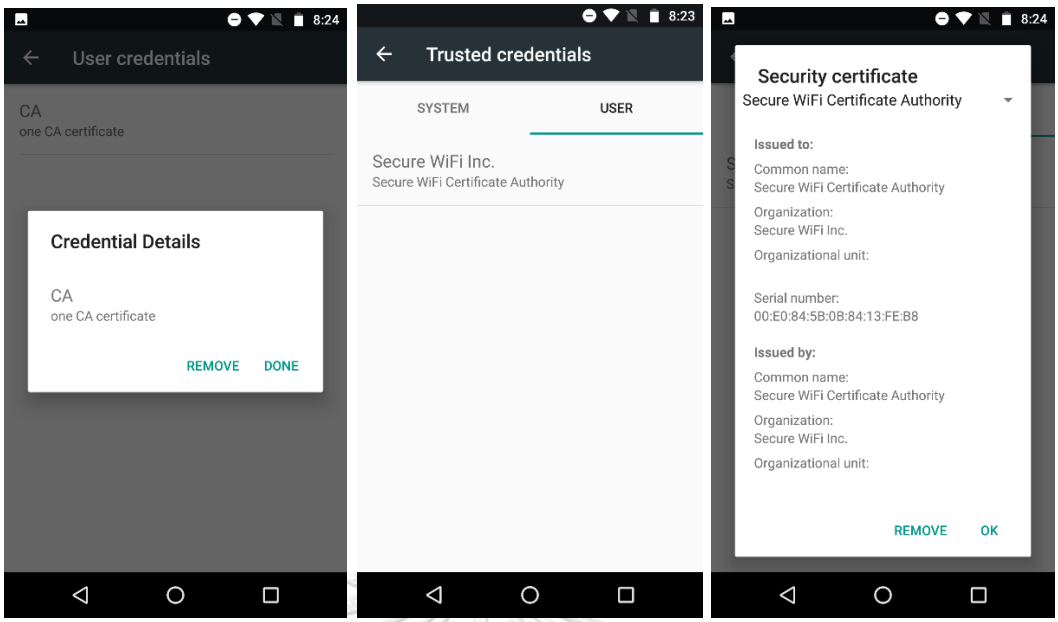
เมื่อแอปพลิเคชันเรียกใช้งานจากทั้งสองฟังก์ชันจะแสดงผลดังภาพที่ 33 โดยภาพ ก. คือภาพขอการยืนยันการติดตั้งใบรับรองที่ออกให้โดยเจ้าหน้าที่ (CA.crt) ซึ่งต้องการลายนิ้วหรือ PIN ยืนยันก่อน และภาพ ข ขอการยืนยันการลงใบรับรองตัวผู้ใช้ซึ่งต้องใช้รหัสผ่านในการปลดล็อคใบรับรอง



(ก) (ข)

ภาพที่ 33 ใบรับรองออกโดยเจ้าหน้าที่ (ก) และใบรับรองตัวผู้ใช้ (ข)

หลังจากผู้ใช้ยืนยันและติดตั้งใบรับรองทั้งสองสำเร็จ ผู้ใช้สามารถเข้าไปตรวจสอบได้ในระบบมือถือดังนี้ Setttings -> Trusted credentials -> USER -> user.p12 (รายละเอียดใบรับรองตัวผู้ใช้) และ Setttings -> User credentials -> CA.crt (รายละเอียดใบรับรองของ CA) ดังภาพที่ 34



ภาพที่ 34 ใบรับรอง CA ติดตั้งสำเร็จ (ก) ใบรับรอง User ติดตั้งสำเร็จ (ข) และรายละเอียด (ค)

นอกจากนี้ส่วนของตัวจัดการใบรับรองยังสามารถช่วยจัดการและป้องกันความเป็นส่วนตัวของผู้ใช้ที่ถูกละเมิด เนื่องจากการติดตั้งใบรับรองโดยไม่มีการยืนยันจากผู้ใช้นั้นอาจนำไปสู่ปัญหาร้ายแรงตามมาได้

4.3.4 ตัวเก็บข้อมูลหนังสือรับรอง (Credential storage)

Keystore เป็นหนึ่งในฟังก์ชันที่ใช้ในการเก็บรวบรวมข้อมูลของระบบ Android โดยข้อมูลที่เก็บรวบรวมส่วนใหญ่นั้นจะเป็นข้อมูลสำคัญ และข้อมูลการเข้ารหัส เช่น ชื่อผู้ใช้ รหัสผ่าน และใบรับรอง เป็นต้น และนอกจากนี้ KeyStore ยังสามารถป้องกันการเข้าถึงข้อมูลโดยไม่ได้รับอนุญาตจากผู้ที่ไม่มีสิทธิ์ (non-rooted) หรือแอปพลิเคชันอื่นในขณะที่ทำงานอยู่ ทั้งนี้หากสมาร์ตโฟนถูกขโมยและผู้ร้ายได้สิทธิ์เป็น root ผู้ร้ายจะสามารถสกัดข้อมูลเหล่านั้นออกมาได้ ดังนั้นถ้าทำให้ข้อมูลสามารถเข้าถึงได้ยากอาจเป็นทางเลือกหนึ่งที่จะช่วยลดความเสี่ยงจากเหตุการณ์ดังกล่าวได้ โดยงานวิจัยนี้ได้นำเอาวิธีการเข้ารหัสแบบไม่สมมาตรเข้ามาใช้ในการเข้ารหัสข้อมูล โดยใช้กุญแจสาธารณะ (Public Key) ของแม่ข่ายก่อนจัดเก็บลงใน Keystore เพื่อความปลอดภัย และข้อมูลดังกล่าวจะถูกถอดรหัสโดยกุญแจส่วนตัว (Private key) ซึ่งมีเพียงแม่ข่ายเท่านั้นที่เก็บไว้

ในส่วนการสร้าง Keystore System สามารถแบ่งออกเป็นสองส่วนหลัก ได้แก่ คลาสการเข้ารหัส (Ecryption class) และ คลาสการถอดรหัส (Decryption class) โดยในงานนี้จะใช้อัลกอริทึม AES ในการเข้ารหัสและถอดรหัสข้อมูล โดยขั้นตอนการพัฒนามีรายละเอียดดังนี้

- คลาสการเข้ารหัส (Ecryption class) ก่อนเริ่มการเข้ารหัสข้อมูลจะต้องมีการกำหนดนามแฝง (alias) ก่อน ซึ่งนามแฝงนี้เป็นตัวแทนชื่อคีย์ของแอปพลิเคชันใน Keystore system จากนั้นประกาศ KeyGenerator เพื่อกำหนดว่าการสร้างคีย์นี้จะใช้อัลกอริทึม AES และต้องการเก็บคีย์หรือข้อมูลความลับนี้ใน Keystore เมื่อกำหนดอินสแตนซ์แล้วจำเป็นต้องสร้าง KeyGenParameterSpec โดยใช้ KeyGenParameterSpec.Builder เพื่อส่งผ่านค่าไปยัง KeyGenerators เมท็อดเริ่มต้น (init) ซึ่ง KeyGenParameterSpec คือคุณสมบัติของคีย์ที่จะถูกสร้างดังภาพที่ 35

```
    @NonNull
    private SecretKey getSecretKey(final String alias) throws NoSuchAlgorithmException,
            NoSuchProviderException, InvalidAlgorithmParameterException {
        final KeyGenerator keyGenerator = KeyGenerator
                .getInstance( KeyProperties.KEY_ALGORITHM_AES, ANDROID_KEY_STORE);
        keyGenerator.init(new KeyGenParameterSpec.Builder(alias,
                purposes: KeyProperties.PURPOSE_ENCRYPT | KeyProperties.PURPOSE_DECRYPT)
                .setBlockModes(KeyProperties.BLOCK_MODE_CBC)
                .setEncryptionPaddings(KeyProperties.ENCRYPTION_PADDING_PKCS7)
                .build());
        return keyGenerator.generateKey();
    }

    byte[] getIv(){
        return iv;
    }

}
```

ภาพที่ 35 รายละเอียดการกำหนดค่า KeyGenerator และ KeyGenParameterSpec

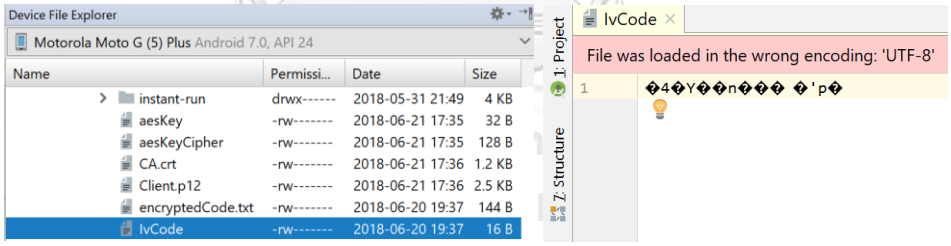
ขั้นตอนต่อมาคือ setBlockModes เพื่อให้มั่นใจได้ว่ามีเพียงโหมดบล็อกที่ระบุไว้เท่านั้นที่สามารถใช้เพื่อเข้ารหัสและถอดรหัสข้อมูล เพราะถ้าหากมีการใช้โหมดบล็อกต่างชนิดหรือไม่เหมือนกัน ระบบจะปฏิเสธและแสดงข้อผิดพลาดในขั้นตอนการถอดรหัสได้ ดังนั้นในส่วนนี้จะใช้บลอคเข้ารหัสและถอดรหัสแบบ “AES/CBC/PKCS7Padding”

เมื่อการตั้งค่าทั้งหมดสำเร็จแล้วต่อไปจะเป็นส่วนการเข้ารหัส โดยเริ่มต้นด้วยกำหนดค่า keyGenerator เริ่มต้นด้วย keyGenParameterSpec หลังจากนั้นสร้างคีย์ลับขึ้นมาแล้ว คีย์นี้จะถูกใช้เพื่อกำหนดค่าเริ่มต้นให้กับ Cipher ซึ่งจะใช้เข้ารหัสจริง โดย Cipher จะได้รับค่าที่จำเป็นในการเข้ารหัสได้แก่ ประเภทของการเข้ารหัสและคีย์ลับ ดังแสดงในภาพที่ 36

```
public class EnCryptor {
    private static final String TRANSFORMATION = "AES/CBC/PKCS7Padding";
    private static final String ANDROID_KEY_STORE = "AndroidKeyStore";

    private byte[] encryption;
    private byte[] iv;

    EnCryptor() {
    }

    byte[] encryptData(final String alias, final byte[] input)
            throws NoSuchAlgorithmException, NoSuchProviderException,
            NoSuchPaddingException, InvalidKeyException, IOException,
            InvalidAlgorithmParameterException, BadPaddingException,
            IllegalBlockSizeException {
        final Cipher cipher = Cipher.getInstance(TRANSFORMATION);
        cipher.init(Cipher.ENCRYPT_MODE, getSecretKey(alias));
        iv = cipher.getIV();
        return (encryption = cipher.doFinal(input));
    }
```

ภาพที่ 36 รายละเอียดฟังก์ชันเข้ารหัส

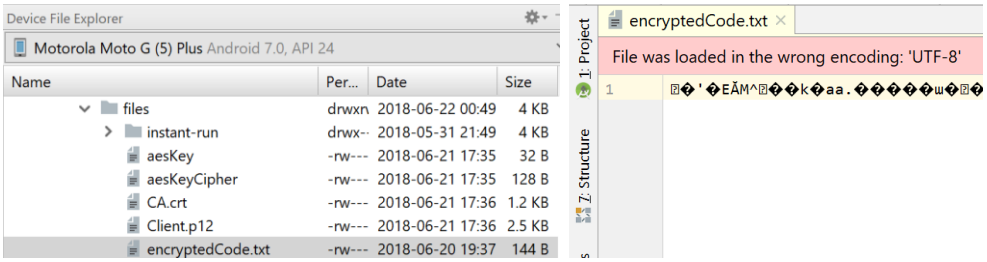
จากนั้นต้องทำการเก็บค่าอ้างอิงถึง ciphers เวกเตอร์การเริ่มต้น (initialization vector หรือ IV) ซึ่ง IV นี้จำเป็นต้องใช้สำหรับการถอดรหัส นั่นคือ IV การเข้ารหัสและถอดรหัสจะต้องเป็นคีย์เดียวกันเท่านั้น ซึ่งทุกครั้งที่กำหนด Cipher เพื่อเข้ารหัสใหม่ IV จะถูกสร้างใหม่แบบสุ่มด้วย โดยเนื้อหา IV จะไม่สามารถอ่านออกได้ดังแสดงในภาพที่ 37 (ก) คือแฟ้มที่เก็บข้อมูลของ IV บนสมาร์ตโฟน และภาพที่ 37 (ข) คือ เนื้อหาของไฟล์ IV



(ก) (ข)

ภาพที่ 37 (ก) คือแฟ้มที่เก็บข้อมูลของ IV บนสมาร์ตโฟน (ข) เนื้อหาของไฟล์ IV

สุดท้ายจะเป็นการเข้ารหัสด้วยคำสั่ง doFinal (input) โดยเมท็อด doFinal ให้ค่าคืนกลับเป็นอาร์เรย์ไบต์ซึ่งเป็นเนื้อหาของการเข้ารหัสข้อมูลจริง โดยแฟ้มที่เก็บข้อมูลของไฟล์เข้ารหัสข้อมูลบนสมาร์ตโฟนแสดงดังภาพที่ 38 (ก) และผลลัพธ์เนื้อหาของไฟล์ที่ถูกเข้ารหัส ดังภาพที่ 38 (ข)



(ก) (ข)

ภาพที่ 38 (ก) แฟ้มเก็บข้อมูลของไฟล์เข้ารหัสบนสมาร์ตโฟน (ข) ผลลัพธ์เนื้อหาของไฟล์ที่ถูกเข้ารหัส

- คลาสการถอดรหัส (Decryption class) ในส่วนของการถอดรหัสข้อมูลจะเริ่มต้นด้วยการกำหนดค่าเริ่มต้นให้กับ KeyStore ก่อนการถอดรหัสซึ่งจำเป็น ต้องมีอินสแตนซ์ของ KeyStore ในที่นี้จะเรียกใช้ “AndroidKeyStore” ดังแสดงในภาพที่ 39

```
package com.wiphop.mynfc;

import ...

public class DeCryptor {

    private static final String TRANSFORMATION = "AES/CBC/PKCS7Padding";
    private static final String ANDROID_KEY_STORE = "AndroidKeyStore";

    private KeyStore keyStore;

    DeCryptor() throws  NoSuchAlgorithmException, KeyStoreException, CertificateException,
            IOException {
            initKeyStore();
    }

    private void initKeyStore() throws KeyStoreException, CertificateException,
            NoSuchAlgorithmException, IOException {
        keyStore = KeyStore.getInstance( ANDROID_KEY_STORE );
        keyStore.load( param: null );
    }

 private SecretKey getSecretKey(final String alias) throws NoSuchAlgorithmException,
         UnrecoverableEntryException, KeyStoreException {
     return ((KeyStore.SecretKeyEntry) keyStore.getEntry(alias, protParam: null)).getSecretKey();
 }
}
```

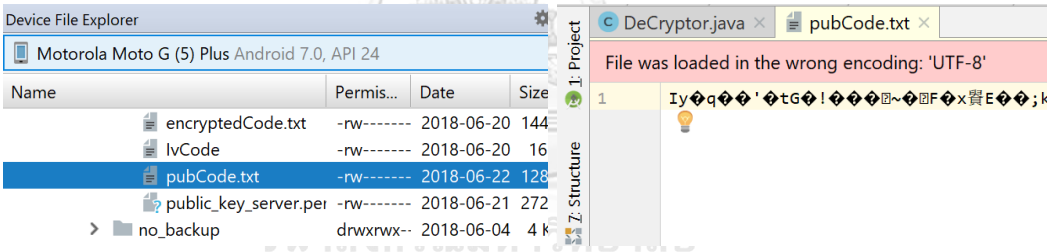
ภาพที่ 39 กำหนดค่าเริ่มต้นให้กับ Keystore system

ขั้นตอนต่อมาคือ การ setBlockModes ซึ่งจะใช้อัลกอริทึมเหมือนกันกับขั้นตอนการเข้ารหัส คือ“AES/CBC/PKCS7Padding” จากนั้นส่งผ่านค่า IV ที่สร้างไว้ก่อนหน้านี้ในขั้นตอนการเข้ารหัส โดยจะใช้ IvParameterSpec เป็นตัวกำหนดค่าเริ่มต้นให้กับกระบวนการถอดรหัส ดังแสดงในภาพที่ 40

```
byte[] decryptData(final String alias, final byte[] encryptedData, final byte[] encryptionIv)
        throws UnrecoverableEntryException, NoSuchAlgorithmException, KeyStoreException,
         NoSuchPaddingException, InvalidKeyException, IOException,
        BadPaddingException, IllegalBlockSizeException, InvalidAlgorithmParameterException {
    final Cipher cipher = Cipher.getInstance( TRANSFORMATION );
    final IvParameterSpec spec = new IvParameterSpec(encryptionIv);
    cipher.init( Cipher.DECRYPT_MODE, getSecretKey( alias ), spec );
    return cipher.doFinal( encryptedData );
}
```

ภาพที่ 40 กำหนด setBlockModes เริ่มต้นให้กับ Keystore

ขั้นตอนท้ายสุดจะดำเนินการถอดรหัสด้วยคำสั่ง doFinal (input) โดยเมท็อด doFinal ให้ค่าคืนกลับเป็นอาร์เรย์ไบต์เช่นเดียวกันกับการเข้ารหัส ซึ่งแฟ้มไฟล์ต้นฉบับถูกจัดเก็บไว้บนสมาร์ตโฟน แสดงดังภาพที่ 41 (ก) และผลลัพธ์เนื้อหาของไฟล์ที่ถูกถอดรหัส ดังภาพที่ 41 (ข)



(ก) (ข)

ภาพที่ 41 (ก) แฟ้มเก็บข้อมูลของไฟล์ถอดรหัสบนสมาร์ตโฟน (ข) ผลลัพธ์เนื้อหาของไฟล์ที่ถูกถอดรหัส

4.3.5 ตัวจัดการเครือข่าย (Network manager)

ฟังก์ชันนี้ช่วยให้สามารถเชื่อมต่อกับ Wireless Access Point (AP) ได้อย่างปลอดภัยโดยไม่ต้องดำเนินการใด ๆ จากผู้ใช้ภายหลังการติดตั้งใบรับรองทั้งสองใบสำเร็จ โดยข้อมูลการกำหนดค่าเครือข่ายนั้นได้รับมาจากเซิร์ฟเวอร์ ได้แก่ ชื่อเครือข่าย (SSID) ประเภทความปลอดภัย (Security type) ชนิด EAP (EAP type) และใบรับรอง (Certificates) ฟังก์ชันจะสร้างโปรไฟล์และกำหนดค่าเครือข่ายให้อัตโนมัติโดยอ้างอิงจากข้อมูลที่ได้รับมาจากฝั่งเครื่องบริการ เช่น ชื่อเครือข่าย (SSID)

ชนิดของ EAP ใช้มาตรฐาน IEEE 802.1X ภาพแบบใบรับรองที่ใช้ จัดเตรียมใบรับรอง และระบุข้อมูลตัวผู้ใช้ เมื่อการตรวจสอบและขั้นตอนการพิสูจน์ตัวจริงสำเร็จ ผู้ใช้จะได้รับสิทธิ์ในการเข้าใช้งานเครือข่ายไร้สายต่อไป

ภาพที่ 42 แสดงถึงการพัฒนาในส่วนของตัวจัดการเครือข่ายแบบอัตโนมัติ ซึ่งเริ่มต้นด้วยการอ่านค่าจากไฟล์กำหนดการตั้งค่าเครือข่าย (network profile) ซึ่งถูกเข้ารหัสไว้ด้วย KeyStore โดยประกอบไปด้วย ตัวระบุหรือชื่อเครือข่ายไร้สาย (SSID), เอกลักษณ์ (Identity), รหัสใบรับรองผู้ใช้, ชนิดความปลอดภัยของเครือข่ายที่ใช้ และ ชนิดของใบรับรอง จากนั้นเมท็อด createWifiProfile จะกำหนดชนิดความปลอดภัยของเครือข่ายในงานนี้ใช้ WPA2-Enterprise และมาตรฐาน IEEE802.1X ซึ่งใช้กรอบงาน EAP-TLS เป็นวิธีการพิสูจน์ตัวจริงนั้นคือใช้ใบรับรองในการตรวจสอบร่วมกัน จากนั้นกำหนดชนิดของใบรับรองที่จะใช้เพื่อเชื่อมต่อเครือข่ายเป้าหมายซึ่ง “X.509” อ้างอิงถึงใบรับรองที่ออกให้โดยเจ้าหน้าที่ Certificate Authority (CA.crt) และ “pkcs12” อ้างอิงถึงใบรับรองตัวผู้ใช้ User Certificate (John.p12)

จากนั้นทำการเปิดอ่านไฟล์ใบรับรองทั้งสองใบเพื่อเตรียมใช้ส่งค่าไปยัง Authentication Server ผ่านทาง Access Point Router ซึ่งใบรับรอง root CA สามารถอ่านไฟล์และสร้างใบรับรองได้ทันที แต่สำหรับในส่วนของใบรับรองตัวผู้ใช้ต้องใช้รหัสผ่านเพื่ออนุญาตให้โปรแกรมสามารถอ่านไฟล์ใบรับรองได้

```
public void createWifiProfile(String ssid, String identity) throws Exception {
    WifiConfiguration config = new WifiConfiguration();
    config.SSID = ssid;
    config.allowedKeyManagement.set( WifiConfiguration.KeyMgmt.WPA_EAP );
    config.allowedKeyManagement.set( WifiConfiguration.KeyMgmt.IEEE8021X );
    config.enterpriseConfig.setEapMethod( WifiEnterpriseConfig.Eap.TLS);
    config.status = WifiConfiguration.Status.ENABLED;
    CertificateFactory certificateGenerator = CertificateFactory.getInstance( "X.509" );
    KeyStore pkcs12ks = KeyStore.getInstance("pkcs12");

    InputStream rootCa = new BufferedInputStream( new FileInputStream(
            CA_CERT ));
    BufferedInputStream in = new BufferedInputStream(new FileInputStream(
            CLIENT_CERT ));

    pkcs12ks.load( in, NETWORK_PWD.toCharArray() );
    X509Certificate caCert = (X509Certificate) certificateGenerator.generateCertificate( rootCa );
```

ภาพที่ 42 ฟังก์ชันผู้ช่วยเชื่อมต่อเครือข่ายเป้าหมาย

ขั้นตอนต่อมาคือกระบวนการอ่านค่าไฟล์ใบรับรองตัวผู้ใช้และกำหนดค่าต่างๆ ดังนี้ กำหนดใบรับรองทั้งสองใบ และ เอกลักษณ์ของผู้ใช้ จากนั้นแอปพลิเคชันทำการเปิดบริการ Wi-Fi ของผู้ใช้

พร้อมทั้งบันทึกข้อมูลของการกำหนดการตั้งค่าไว้บนระบบมือถือเพื่อใช้เชื่อมต่อเองอัตโนมัติในครั้งถัดไป ท้ายสุดแอปพลิเคชันส่งข้อมูลทั้งหมดไปยัง Access Point เพื่อขอตรวจสอบและพิสูจน์ตัวจริงกับ Authentication server รายละเอียดดังแสดงในภาพที่ 43

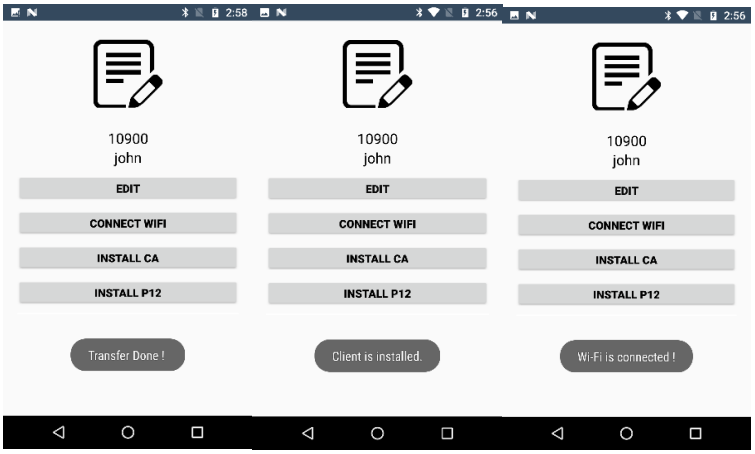
```
    Enumeration<String> aliases = pkcs12ks.aliases();
    while(aliases.hasMoreElements()){
        String alias = aliases.nextElement();

        X509Certificate cert = (X509Certificate) pkcs12ks.getCertificate(alias);

        PrivateKey key = (PrivateKey) pkcs12ks.getKey(alias, NETWORK_PWD.toCharArray());

        config.enterpriseConfig.setClientKeyEntry(key, cert);
        config.enterpriseConfig.setCaCertificate( caCert );
        config.enterpriseConfig.setIdentity(identity);
    }
    Context context = getApplicationContext();
    WifiManager wifiManager = (WifiManager) context.getSystemService(getApplicationContext().WIFI_SERVICE);
    int netID = wifiManager.addNetwork(config);
    wifiManager.saveConfiguration();
    wifiManager.enableNetwork(netID,  attemptConnect: true);
    mcheck = checkWifiOnAndConnected();
}
```

ภาพที่ 43 ฟังก์ชันผู้ช่วยเชื่อมต่อเครือข่ายเป้าหมาย

4.3.6 ฟังก์ชันที่จำเป็นในแอปพลิเคชัน

4.3.6.1 ฟังก์ชันแจ้งเตือน

ในฟังก์ชันนี้มีสองชนิดหลักๆ ได้แก่ 1) แจ้งเตือนผู้ใช้ด้วยการสั่น และ 2) แจ้งเตือนผู้ใช้ด้วยข้อความ pop-up การแจ้งเตือนเป็นส่วนหนึ่งที่สำคัญในการสื่อสารระหว่างแอปพลิเคชันและผู้ใช้งาน ตัวอย่างสถานการณ์ในงานนี้เช่น หากผู้ใช้แตะสมาร์ตโฟนเพื่อรับใบรองสำเร็จ สถานะ Wi-Fi เชื่อมต่อสำเร็จ และการติดตั้งใบรับรองสำเร็จจะมีการแจ้งเตือนด้วยการสั่นและข้อความดังภาพที่ 44



ภาพที่ 44 ตัวอย่างการแจ้งเตือน

เมท็อดตัวที่กำหนดไว้สองแบบคือ แจ้งเตือนด้วยการสั่นดังภาพที่ 45 และแจ้งเตือนด้วยข้อความดังภาพที่ 46

```
private void vibrate(){
    Vibrator v = (Vibrator) getSystemService(Context.VIBRATOR_SERVICE );

    //Vibrate for 3 seconds
    if(Build.VERSION.SDK_INT >= Build.VERSION_CODES.O){
        v.vibrate( VibrationEffect.createOneShot( milliseconds: 300,VibrationEffect.DEFAULT_AMPLITUDE ) );
    }else{
        v.vibrate( milliseconds: 300 );
    }
}
```

ภาพที่ 45 ตัวอย่างเมท็อดแจ้งเตือนผู้ใช้งานด้วยการสั่น

```
private void notificationAlert()
    {
        tEnd = System.currentTimeMillis();
        tDelta = tEnd - tStart;
        elapsedSeconds = tDelta / 1000.0;
        Log.d( tag: "TIME", msg: "Time completed to exchange data: " + elapsedSeconds + " Seconds");
        vibrate();
        Toast.makeText( context: this, text: "Transfer Done ! ",Toast.LENGTH_SHORT).show();
    }

private void errorNotificationAlert()
{
    vibrate();
    Toast.makeText( context: this, text: "User does not exist ! ",Toast.LENGTH_SHORT).show();
}
```

ภาพที่ 46 ตัวอย่างเมท็อดแจ้งเตือนผู้ใช้งานด้วย pop-up

4.3.6.2 ฟังก์ชันจัดการรูปแบบใบรับรอง

ฟังก์นี้ทำหน้าที่แปลงรูปแบบใบรับรองที่รับมาจากเซิร์ฟเวอร์ซึ่งตัวอย่างเนื้อหาของใบรับรองดังแสดงในภาพที่ 47 ให้สามารถเข้ากันกับรูปแบบของระบบปฏิบัติการแอนดรอยด์ โดยลบบรรทัดเริ่มต้นและบรรทัดสุดท้ายของใบรับรองออกและเลือกเฉพาะส่วนของเนื้อหาใบรับรอง โดยมีสองตัวอย่างเมท็อดที่ใช้ประกอบด้วย ตัวจัดการใบรับรองสำหรับกุญแจส่วนตัวดังแสดงตัวอย่างในภาพที่ 48 และตัวจัดการใบรับรองสำหรับกุญแจสาธารณะดังแสดงตัวอย่างในภาพที่ 49

```
-----BEGIN PRIVATE KEY-----
MIICdgIBADANBgkqhkiG9w0BAQEFAASCAmAwggJcAgEAAoGBAM6MeafwyMMgCc1Z
rBj/gczPzutcZdLKkV3BohTD1YqFBuhwJIs4bANdIFR7O+TMl6/jHTEl+Sq6bBtA
LdOxsFmkP3P8HUYyMwcQl6X/LNzf7jzj1PwUM/wC5re3iqyqzCBMHThRElDSBDX/
ETroVyXNneqKT1C1KRADWpq1A4RJAgMBAAECgYAZWG+NRiS8nO/W/Qv7PU9GPLHd
e8ca6LSR1rL27VkE2HISj8BMPS4z83eGJmEhIGH0uV/Zqhk7PwkJBwUYlxNWlDqi
pU59GBqjI3IRQgk12LcLwq9rCpm6Qo3ORp1rM1jXPPYC9DZy6FvPu9bUhMLnDmnl
nkO8cPOUJG1IWFjuqQJBAOj13NTXrQvdK7v/uWK0jX7kcs9Qbj70mKZ0K3WWbTZ1
1gxdsBVhcNTpP3fv2Q+Cd9YvRMxOmhjc69jIiyFr6JMCQQDi+eqvShsSm4f9RPIE
/oVvBCvxZoMzQehJSC8Jnp+Quw3mcvHpQ0VBVQXyMsBnVUpJqHR1svR4xXQRDxNo
g3UzAkA0yGg1rXxe4Mjk+scFT0V0B8zfniisiP0BUfb62I98NGzANoYMAhzvkL25
rhVvRy9hzyn2ZKTTl1K0tpOXifyTAkAXmWB7RTs3XhBvluGK/2OkPUzMVppBlBKw
u96YI3P3ulshoMXiOOpUdqCdpAU5OTb2HZuuOSMOA4mj1GtyLGjdAkEAgu6Dow0Z
Qz3D3lNCaaaZW32MvZ6Oi+laEKwB1Lv/4GtuVJdDUZ7Ai63RwkSQxtLWbuy7DTKe
kEqop/YYwbZaHw==
-----END PRIVATE KEY-----
```

ภาพที่ 47 ตัวอย่างเนื้อหาใบรับรองเดิมที่ได้รับจากฝั่งเซิร์ฟเวอร์

```
private static String getPrivateKey(String filename) throws IOException
{
    // Read key from file
    String strKeyPEM = "";
    BufferedReader br = new BufferedReader(new FileReader(filename));
    String line;
    while ((line = br.readLine()) != null)
        {
            strKeyPEM += line + "\n";
        }
    strKeyPEM = strKeyPEM.replace( target: "-----BEGIN PRIVATE KEY-----\n", replacement: "" );
    strKeyPEM = strKeyPEM.replace( target: "-----END PRIVATE KEY-----\n", replacement: "" );
    br.close();
return strKeyPEM;
}
```

ภาพที่ 48 เมท็อดสำหรับแปลงรูปแบบใบรับรองกุญแจส่วนตัว


```
private static String getPublicKey(String filename) throws IOException {
    // Read key from file
    String strKeyPEM = "";
    BufferedReader br = new BufferedReader(new FileReader(filename));
    String line;
    while ((line = br.readLine()) != null) {
        strKeyPEM += line + "\n";
    }
    strKeyPEM = strKeyPEM.replace( target: "-----BEGIN PUBLIC KEY-----\n", replacement: "");
    strKeyPEM = strKeyPEM.replace( target: "-----END PUBLIC KEY-----", replacement: "");
    Log.d( tag: "HCE", strKeyPEM );
    br.close();
    return strKeyPEM;
}
```

ภาพที่ 49 เมท็อดสำหรับแปลงรูปแบบใบรับรองสาธารณะ

ภาพที่ 50 แสดงถึงตัวอย่างผลลัพธ์ของใบรับรองที่ได้จากการผ่านเมท็อดดังกล่าว

```
MIICdgIBADANBgkqhkiG9w0BAQEFAASCAmAwggJcAgEAAoGBAM6MeafwyMMgCc1Z
rBj/gczPzutcZdLKkV3BohTD1YqFBuhwJIs4bANdIFR7O+TMl6/jHTEl+Sq6bBtA
LdOxsFmkP3P8HUYyMwcQl6X/LNzf7jzj1PwUM/wC5re3iqyqzCBMHThRElDSBDX/
ETroVyXNneqKT1C1KRADWpq1A4RJAgMBAAECgYAZWG+NRiS8nO/W/Qv7PU9GPLHd
e8ca6LSR1rL27VkE2HISj8BMPS4z83eGJmEhIGH0uV/Zqhk7PwkJBwUYlxNWlDqi
pU59GBqjI3IRQgk12LcLwq9rCpm6Qo3ORp1rM1jXPPYC9DZy6FvPu9bUhMLnDmnl
nkO8cPOUJG1IWFjuqQJBAOj13NTXrQvdK7v/uWK0jX7kcs9Qbj70mKZ0K3WWbTZ1
1gxdsBVhcNTpP3fv2Q+Cd9YvRMxOmhjc69jIiyFr6JMCQQDi+eqvShsSm4f9RPIE
/oVvBCvxZoMzQehJSC8Jnp+Quw3mcvHpQ0VBVQXyMsBnVUpJqHR1svR4xXQRDxNo
g3UzAkA0yGg1rXxe4Mjk+scFT0V0B8zfniisiP0BUfb62I98NGzANoYMAhzvkL25
rhVvRy9hzyn2ZKTTl1K0tpOXifyTAkAXmWB7RTs3XhBvluGK/2OkPUzMVppBlBKw
u96YI3P3ulshoMXiOOpUdqCdpAU5OTb2HZuuOSMOA4mj1GtyLGjdAkEAgu6Dow0Z
Qz3D3lNCaaaZW32MvZ6Oi+laEKwB1Lv/4GtuVJdDUZ7Ai63RwkSQxtLWbuy7DTKe
kEqop/YYwbZaHw==
```

ภาพที่ 50 ผลลัพธ์ของเนื้อหาใบรองเมื่อผ่านเมท็อดจัดการรูปแบบใบรับรอง

4.3.6.3 ฟังก์ชันเข้ารหัสข้อมูลส่วนตัวของผู้ใช้

ฟังก์ชันนี้ทำหน้าที่เข้ารหัสข้อมูลส่วนตัวของผู้ใช้ด้วยใบรับรองกุญแจสาธารณะของเซิร์ฟเวอร์ ดังภาพที่ 51 ซึ่งกระบวนการนี้จะสามารถทำได้ก็ต่อเมื่อผู้ใช้ได้รับใบรับรองกุญแจสาธารณะผ่านช่องทางเอ็นเอฟซี ฟังก์ชันนี้ถูกพัฒนาเพื่อป้องกันไม่ให้ข้อมูลผู้ใช้ถูกเปิดเผย โดยข้อมูลที่ถูกเข้ารหัสดังกล่าวจะมีเพียงฝั่งเซิร์ฟเวอร์เท่านั้นที่สามารถถอดรหัสได้ด้วยกุญแจส่วนตัวของเซิร์ฟเวอร์

```
protected void savePubCode(File in, File out) throws IOException, GeneralSecurityException {
    String publicKey = getPublicKey( PUBLIC_KEY_SEVER_FILE );
    byte[] encodedKey = android.util.Base64.decode( publicKey.trim(), android.util.Base64.NO_WRAP );
    //create public key
    KeyFactory kf = KeyFactory.getInstance( algorithm: "RSA", mProvider);
    RSAPublicKey pk = (RSAPublicKey) kf.generatePublic(new X509EncodedKeySpec(encodedKey));
    pkCipher.init( Cipher.ENCRYPT_MODE, pk );
    FileInputStream is = new FileInputStream(in);
    CipherOutputStream os = new CipherOutputStream( new FileOutputStream( out ), pkCipher );
    copy(is, os);
    is.close();
    os.close();
    detleteCode();
}
```

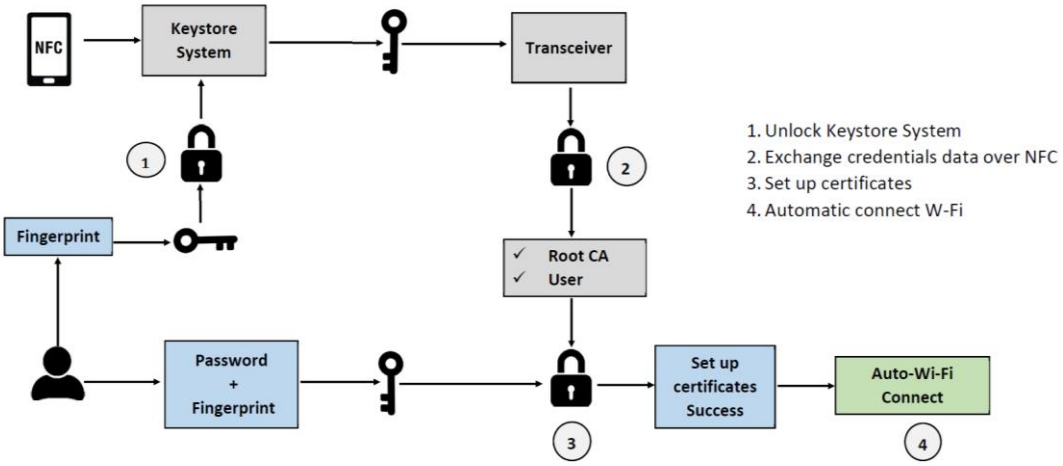
ภาพที่ 51 ฟังก์ชันเข้ารหัสข้อมูลส่วนตัวของผู้ใช้

4.4 ขั้นตอนการทำงาน

4.4.1 ขั้นตอนการทำงานของการพิสูจน์ตัวจริง

ในงานนี้เอ็นเอฟซีถูกนำมาใช้เพื่อเป็นช่องทางในการแลกเปลี่ยนใบรับรองและข้อมูลการตั้งค่าเครือข่าย ซึ่งใช้การพิสูจน์ตัวจริงแบบหลายปัจจัย (MFA) เพื่อเพิ่มความปลอดภัยให้มากยิ่งขึ้น

กระบวนการทำงานของขั้นตอนการพิสูจน์ตัวจริงเมื่อผู้ใช้เชื่อมต่อกับ Wi-Fi hotspot ขององค์กรมีรายละเอียดดังแสดงในภาพที่ 52 ซึ่งแบ่งออกเป็น 4 ขั้นตอนดังนี้ 1) ปลดล็อกระบบ Keystore ของผู้ใช้ 2) แลกเปลี่ยนข้อมูลข้อมูลใบรับรองผ่านเอ็นเอฟซี 3) ติดตั้งใบรับรอง และ 4) เชื่อมต่อเครือข่ายไร้สายอัตโนมัติ



ภาพที่ 52 ขั้นตอนการทำงานของการพิสูจน์ตัวจริง

- ปลดล็อกระบบ Keystore ของผู้ใช้

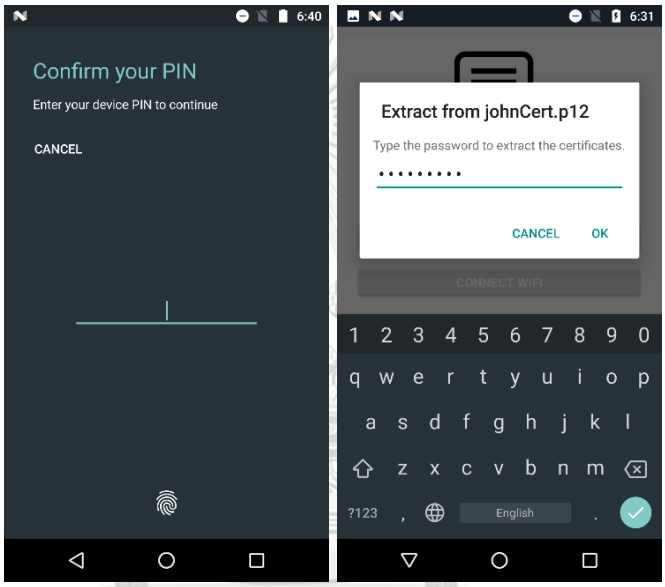
เริ่มต้นด้วยผู้ใช้แตะลายนิ้วมือบนตัวรับรู้ (sensor) ของอุปกรณ์เพื่อปลดล็อกระบบ Keystore บนโทรศัพท์ ขั้นตอนนี้ถือว่าเป็นปัจจัยแรกของกระบวนการพิสูจน์ตัวจริงแสดงให้เห็นถึง "สิ่งที่คุณเป็น" หรือ “what you are”

- แลกเปลี่ยนข้อมูลใบรับรองผ่านเอ็นเอฟซี

ขั้นตอนต่อมาคือผู้ใช้นำสมาร์ตโฟนแตะที่ตัวอ่านเอ็นเอฟซี จากนั้นโปรแกรมเอ็นเอฟซีที่ทำงานอยู่บนอยู่ฝั่งเครื่องบริการ จะสอบถามเลขประจำตัวแอปพลิเคชัน (AID) บนอุปกรณ์เป้าหมาย โดยโปรแกรมจะตรวจสอบว่าข้อมูลผู้ใช้มีอยู่ในฐานข้อมูลหรือไม่ เมื่อผลการตรวจสอบถูกต้อง ข้อมูลใบรับรอง CA.crt และใบรับรองผู้ใช้ User.p12 จะได้รับการเข้ารหัสและส่งกลับไปยังสมาร์ตโฟน ในขั้นตอนนี้ข้อมูลผู้ใช้ที่จัดเก็บด้วยระบบ Keystore ได้ถูกใช้สำหรับยืนยันสิทธิ์กับเซิร์ฟเวอร์ซึ่งถือเป็นปัจจัยที่สองคือ "สิ่งที่คุณมี" หรือ “what you have”

- ติดตั้งใบรับรอง

เมื่อเซิร์ฟเวอร์ส่งใบรับรองและข้อมูลการตั้งค่าเครือข่ายแล้ว แอปพลิเคชันบนสมาร์ตโฟนจะเรียกฟังก์ชันติดตั้งใบรับรองพร้อมการแจ้งเตือนผู้ใช้เพื่อยืนยันการดำเนินการนำเข้าใบรับรอง ขั้นตอนนี้สำคัญต่อความเป็นส่วนตัวผู้ใช้ เนื่องจากผู้ใช้ต้องทราบและยืนยันก่อนติดตั้งใบรับรองลงในอุปกรณ์ของตน โดยขั้นตอนนี้ต้องใช้ลายนิ้วมือ หรือ รหัส PIN เพื่อตั้งค่า CA.crt ดังภาพที่ 53 (ก) และใช้รหัสผ่านเพื่อติดตั้งใบรับรองตัวผู้ใช้ johnCert.p12 ดังภาพที่ 53 (ข) ซึ่งขั้นตอนนี้แสดงถึงปัจจัย “สิ่งที่คุณรู้” และ ”สิ่งที่คุณเป็น” หรือ “What you know” และ “What you are”



(ก) (ข)

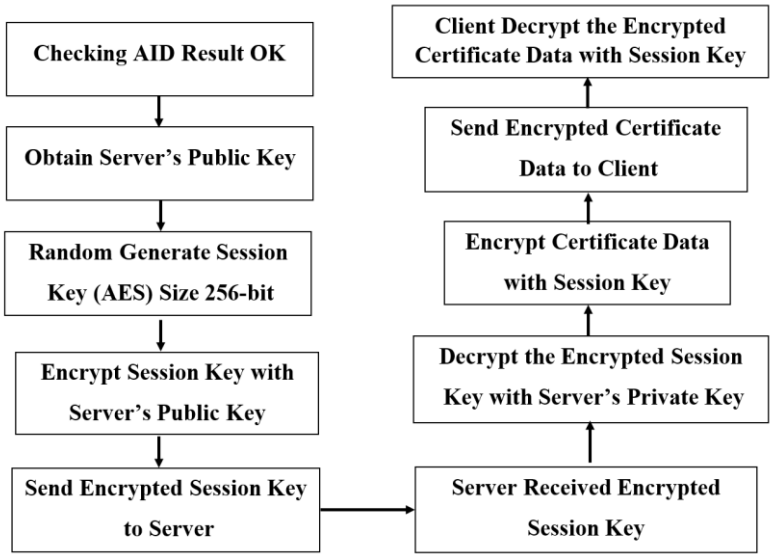
ภาพที่ 53 การติดตั้งใบรับรอง CA.crt (ก) และการติดตั้งใบรับรองผู้ใช้ johnCert.p12 (ข)

- เชื่อมต่อ Wi-Fi โดยอัตโนมัติ

ในขั้นตอนสุดท้ายฟังก์ชันการจัดการเครือข่ายบังคับให้อุปกรณ์เชื่อมต่ออย่างปลอดภัยกับจุดเชื่อมต่อไร้สายเป้าหมายด้วยข้อมูลเครือข่ายก่อนที่ได้รับจากฝั่งเซิร์ฟเวอร์รวมถึงตัวระบุชุดบริการ (SSID) ประเภทความปลอดภัยวิธี EAP ใบรับรอง และข้อมูลประจำตัวผู้ใช้ ซึ่งข้อดีของวิธีการนี้คือช่วยลดความเสี่ยงที่ผู้ใช้เชื่อมต่อกับจุดพร้อมโยงที่ไม่ปลอดภัยและลดความผิดพลาดจากการเลือกใช้ชื่อเครือข่ายเป้าหมายที่ไม่ถูกต้อง (Evil Twin Attack)

4.4.2 ขั้นตอนการทำงานของการรับส่งข้อมูลใบรับรอง

การส่งและรับข้อมูลระหว่างสมาร์ตโฟนและเครื่องบริการเริ่มต้นหลังจากคำสั่ง SELECT_AID ตรวจสอบสำเร็จ รูปแบบของกระบวนการแลกเปลี่ยนข้อมูลดังกล่าวจะเป็นแบบต่อเนื่อง นั่นคือจะสิ้นสุดก็ต่อเมื่อผู้ใช้ยกเลิกการแตะสามาร์ทโฟนกับตัวอ่าน ดังนั้นจึงต้องมีการกำหนดเงื่อนไขเพิ่มเติมเพื่อคัดกรองเนื้อหาข้อมูลด้วยการนับรอบที่ได้รับข้อมูลจากฝั่งเครื่องบริการ ซึ่งมีลำดับและรายละเอียดดังภาพที่ 54



ภาพที่ 54 ขั้นตอนการแลกเปลี่ยนข้อมูล

ภาพที่ 55 เป็นเงื่อนไขเพื่อตรวจสอบว่าฝั่งเครื่องบริการได้ส่งใบรับรองกุญแจสาธารณะ ซึ่งจะเขียนเนื้อหาใบรับรองนี้ลงในแฟ้มของแอปพลิเคชันแบบโหมดส่วนตัว (private write mode) กล่าวคือแอปพลิเคชันนี้เท่านั้นที่สามารถอ่าน เขียน และกระทำการได้ สำหรับขั้นตอนนี้ใช้ทั้งหมด 3 รอบเพื่อรับไฟล์ใบรับรองจนสำเร็จ

```
processApdu:
    try
        {
          // Log.d("HCE", "TEST_1");
          if (apdu != temApdu && count < 4 && !Arrays.equals( DictInf.RClear,apdu)){
              Log.d( tag: "HCE", msg: "Yes..pkCipher loop is receiveing....." + count++ );
              FileUtils.writeByteArrayToFile( mFilePkServer, apdu, append: true );
              temApdu = apdu;
          }
```

ภาพที่ 55 เงื่อนไขตรวจสอบและเขียนไฟล์ใบรับรองจากฝั่งเครื่องบริการ

ภาพที่ 56 คือเมท็อดที่โปรแกรมบนเครื่องบริการใช้ในการส่งใบรับรองกุญแจสาธารณะให้กับสมาร์ตโฟน

```
private void sendServerPk() throws IOException, CardException, GeneralSecurityException {

    System.out.print("\n-----------------PK SERVER TRANFER BEGIN-------------------\n");

    //To send server pk cipher to client
    Path pathPkServer = Paths.get(PUBLIC_KEY_SERVER_FILE);
    byte[] msg_pkServer = Files.readAllBytes(pathPkServer);
    sendMessage(msg_pkServer);
    responseAPDU(DictInf.RTransferClientCert);

    System.out.print("-----------------PK SERVER TRANFER END-------------------\n");

}
```

ภาพที่ 56 เครื่องบริการส่งใบรับรองกุญแจสาธารณะให้กับสามาร์ทโฟน

เมื่อรันโปรแกรม ผลลัพธ์ที่ได้จากทางหน้าจอของฝั่งเครื่องบริการดังแสดงในภาพที่ 57

```
Main.java  Communicator.java  Console  DictInf.java
<terminated> Main (2) [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x86_64/bin/java (Jun 22, 2018, 12:35:37 AM)

input: 0x00 0xA4 0x04 0x00 0x07 0xF0 0x01 0x02 0x03 0x04 0x05 0x06 0x00
Selection successful. Connection established!
Response after AID Selection: AIDSelectionOk

checkIfReady ReaderResponse message: waitPlease

-----------------PK SERVER TRANFER BEGIN-------------------

-----------------START--------------------------
sendBackMessage: -----BEGIN PUBLIC KEY-----
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDOjHmn8MjDIAnNWawY/4HMz87r
XGXSypFdwaIUw9WKhQbocCSLOGwD

sendBackMessage as Byte:
0xFF 0x00 0x00 0x00 0x7A 0xD4 0x40 0x01 0x2D 0x2D 0x2D 0x2D 0x2D 0x42 0x45 0x47 0x49 0x4E 0x20 0x50
-----------------END--------------------------

0

-----------------START--------------------------
sendBackMessage: XSBUezvkzJev4x0xJfkqumwbQC3TsbBZpD9z
/B1GMjMHEJel/yzc3+4849T8FDP8Aua3t4qsqswgTB04URJQ0gQ1/xE66FclzZ3q
ik9QtSkQA1qatQOESQ
```

ภาพที่ 57 เครื่องบริการส่งใบรับรองกุญแจสาธารณะให้กับสามาร์ตโฟน

ภาพที่ 58 คือเงื่อนไขที่โปรแกรมฝั่งสมาร์ตโฟนใช้ตรวจสอบหากฝั่งเครื่องบริการส่งคำสั่งขอแลกเปลี่ยนข้อมูลของผู้ใช้ ซึ่งแอปพลิเคชันจะเรียกฟังก์ชัน sendMessage ที่จะจัดเตรียม session key เพื่อใช้เข้ารหัสข้อมูลที่จะถูกขนส่งระหว่างสมาร์ตโฟนและเครื่องบริการ โดย session key นี้เป็นการเข้ารหัสแบบสมมาตรและเป็นคีย์ที่ผู้ส่งและผู้รับใช้ในการเข้ารหัสและถอดรหัสด้วยคีย์เดียวกัน ดังนั้นก่อนการส่งข้อมูลผู้ใช้ออกจากอุปกรณ์ไปยังเครื่องบริการ session key จะต้องถูกส่งไปยัง

เครื่องบริการอย่างปลอดภัยด้วยการเข้ารหัส session key ด้วยใบรับรองกุญแจสาธารณะที่ได้รับจากเงื่อนไขก่อนหน้า

```
else if(apdu != temApdu && count > 3  && count < 5 && Arrays.equals(DATA_EXCHANGE_ORDER,apdu)){
    Log.d( tag: "HCE", msg: "Yes..sendAesCipher loop is receiveing....." + count++);
    temApdu = apdu;
    return sendMessage();
}
```

ภาพที่ 58 เครื่องบริการส่งคำสั่งเพื่อขอแลกเปลี่ยนข้อมูลผู้ใช้

ภาพที่ 59 คือเมท็อดของโปรแกรมทางฝั่งเครื่องบริการที่ส่งคำสั่งขอเริ่มต้นการแลกเปลี่ยนข้อมูลผู้ใช้ ซึ่งเครื่องบริการจะได้รับ session key ทำการเขียนบันทึกไฟล์ session key และถอดรหัสเพื่อใช้คีย์เข้ารหัสข้อมูลใบรับรองที่จะถูกแลกเปลี่ยนในขั้นตอนอื่นถัดไป

```
private void askForSessionKey() throws CardException, InterruptedException, IOException, GeneralSecurityException
{
    try
        {
            ResponseAPDU readerResponseStatus  = channel.transmit(new CommandAPDU(DATA_EXCHANGE_ORDER)); //send e
            String s = new String(readerResponseStatus.getBytes(), "US-ASCII");
            System.out.println("\naskForSessionKey ReaderResponse message: " + s);
            mFileAesCipher.delete();
            //FileUtils.writeByteArrayToFile( mFileAesCipher, Base64.decodeBase64(s.getBytes()), true );
            FileUtils.writeByteArrayToFile( mFileAesCipher, readerResponseStatus.getBytes(), true );
            //communicate();
        }
    catch(Exception e)
        {
            e.printStackTrace();
            System.out.println("\nSend command error: " + e);
            reConnect();
        }
}
```

ภาพที่ 59 เครื่องบริการส่งคำสั่งเพื่อเริ่มขอแลกเปลี่ยนข้อมูล

หลังจากที่ทั้งฝั่งสมาร์ตโฟนและฝั่งเครื่องบริการมี session key ทั้งคู่แล้วจะทำการส่งข้อมูลผู้ใช้จากอุปกรณ์มือถือไปยังเครื่องบริการ โดยเนื้อหาจะถูกเข้ารหัสด้วย session key ทั้งหมดดังแสดงในภาพที่ 60 และภาพที่ 61 แสดงถึงเมท็อดของโปรแกรมเครื่องบริการได้ส่งข้อความขอข้อมูลผู้ใช้ซึ่งประกอบด้วย ชื่อ และรหัสประจำตัว จากอุปกรณ์เป้าหมาย

```
else if(count > 4) {
    try {
            if (count < 6 && Arrays.equals( encryptMsg( DictInf.RAskForUser), apdu ))
                {
                    Log.d( tag: "HCE", msg: "Yes..AskForUser loop is receiveing....." + count++);
                    return reposeAskForUser();
                }
```

ภาพที่ 60 สมาร์ตโฟนตอบกลับข้อมูลผู้ใช้ไปยังเครื่องบริการ

```
byte[] preparedMessage = getMessage(encryptMsg(DictInf.RAskForUser));
readerResponse = channel.transmit(new CommandAPDU(preparedMessage));
readerResponseBytes = processResponse(readerResponse.getBytes());
```

ภาพที่ 61 เครื่องบริการส่งข้อความเพื่อขอข้อมูลผู้ใช้ไปยังอุปกรณ์

เมื่อเซิร์ฟเวอร์ได้รับข้อมูลผู้ใช้แล้ว โปรแกรมจะตรวจสอบข้อมูลที่ได้รับกับฐานข้อมูลของผู้ใช้ผ่านทางคลาส JDBC API ซึ่งจาวาใช้เพื่อเป็น database API ดังภาพที่ 62 หากพบจะทำการส่งไฟล์ใบรับรอง CA และใบรับรองตัวผู้ใช้ user.p12 ไปยังสมาร์ตโฟนดังภาพที่ 63

```
public boolean checkCredential(String mName, String mId ) throws SQLException, IOException
    {
        mCheck = false;
        sendBackMessage = null;
        Connection Conn = DriverManager.getConnection(DB_URL, USER, PASS);

        try
            {
                Class.forName(JDBC_DRIVER);
                System.out.println("\nDriver loaded!");

            }
        catch (ClassNotFoundException e)
            {
                throw new IllegalStateException("Cannot find the driver in the classpath!", e);
            }

        System.out.println("Connecting database......");
        Statement Stmt = (Statement) Conn.createStatement();
        try(Connection connection = DriverManager.getConnection(DB_URL, USER, PASS)){

            System.out.println("Database connected!\n");

ResultSet resultSetcheckId = Stmt.executeQuery("SELECT * FROM radcheck where '"+mId+"'");

PreparedStatement resultSetNewUser = (PreparedStatement)
       Conn.prepareStatement("INSERT INTO radcheck (id, username, attribute, op, value) "
               + "VALUES ('"+1+"','"+mId+"','"+"User-Password"+"','"+":="+"', '"+tmpPwd+"')");

while(resultSetcheckId.next())
    {
            if(resultSetcheckId.getString(2) != null && resultSetcheckId.getString(2).equals(mId))
                {
                    System.out.println("Yes...found the existed user: " + resultSetcheckId.getString(2));
                    return mCheck = true;
                }
            else
                {
                    System.out.println("No....cannot found the entry: " + resultSetcheckId.getString(2));
                    continue;
                }
    }
```

ภาพที่ 62 เมท็อด JDBC ในจาวาช่วยในการตรวจสอบผู้ใช้กับฐานข้อมูล

```
//Prepare Certificates
prepareCert();


Path pathCa = Paths.get(ENCRYPTED_CA_CERT);
byte[] msg_ca = Files.readAllBytes(pathCa); //convert file byte
sendMessage(msg_ca);   //Transmit data to device
responseAPDU(encryptMsg(DictInf.RTransferCACert));
System.out.print("-----------------CA TRANFER END-------------------\n");


Path pathClient = Paths.get(ENCRYPTED_CLIENT_CERT);
byte[] msg_client = Files.readAllBytes(pathClient);
sendMessage(msg_client);
responseAPDU(encryptMsg(DictInf.RTransferClientCert));
System.out.print("-----------------CLIENT TRANFER END-------------------\n");


responseAPDU(encryptMsg(DictInf.RStopTransfer));
System.out.print("-----------------DATA TRANFER END-------------------");
transferDone();
reConnect();
```

ภาพที่ 63 เครื่องบริการส่งเนื้อหาใบรับรองกลับไปยังสมาร์ตโฟน

ภาพที่ 64 แสดงให้เห็นถึงผลลัพธ์ของโปรแกรมตรวจสอบผู้ใช้กับฐานข้อมูลบนเครื่องบริการ

```
Loop decryptUserCode 7 Iy@q@@'@tG@!@@@@~@@F@x巽E@@;k001?粟ø@dOC.@@o[@@ O@@T@@W@@@u[

Decrypt of byteUserCode: John 1629900178036

User ID: 1629900178036
Name: John

Driver loaded!
Connecting database......
Database connected!

No....cannot found the entry: wiphop
No....cannot found the entry: mild
Yes...found the existed user: 1629900178036

Statement & Database connection is closed...
```

ภาพที่ 64 ผลลัพธ์ทางหน้าจอของการตรวจสอบข้อมูลผู้ใช้กับฐานข้อมูล

ภาพที่ 65 คือเงื่อนไขที่รอรับข้อมูลเนื้อหาใบรับรอง CA และ ภาพที่ 66 คือเงื่อนไขที่รอรับข้อมูลเนื้อหาใบรับรองผู้ใช้ ซึ่งข้อมูลทั้งหมดจะถูกเขียนบันทึกไว้ในแฟ้มของแอปพลิเคชัน

```
if ( !Arrays.equals(encryptMsg(DictInf.RUserNotFound), apdu ))
    {
        if((apdu != temApdu) && count < 17 && !Arrays.equals( encryptMsg( DictInf.RAskForUser), apdu ))
            {
                if ( Arrays.equals(encryptMsg(DictInf.RTransferCACert), apdu ))
                    {
                        break processApdu;
                    }
                Log.d( tag: "HCE",  msg: "Yes..CACert loop is receiveing....." + count++ );
                FileUtils.writeByteArrayToFile( mFileCAEncrpted, apdu,  append: true );
                temApdu = apdu;
            }
```

ภาพที่ 65 เงื่อนไขเพื่อตรวจสอบและรับเนื้อหาไฟล์ใบรับรอง CA

```
else if((apdu != temApdu) && count < 40 && !Arrays.equals( encryptMsg( DictInf.RTransferCACert), apdu ))
    {
        if (Arrays.equals( encryptMsg(DictInf.RTransferClientCert), apdu ) ||
                Arrays.equals( encryptMsg(DictInf.RTransferCACert), apdu ))
        {
            break processApdu;
        }
        FileUtils.writeByteArrayToFile( mFileClientEncrypted, apdu,  append: true );
        temApdu = apdu;
    }
```

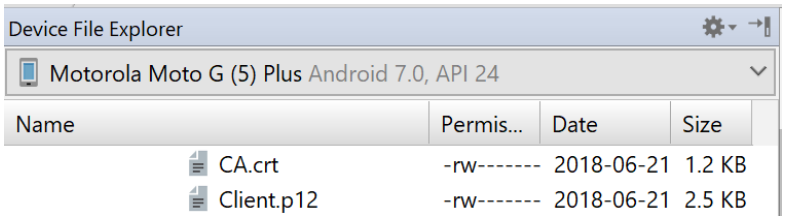
ภาพที่ 66 เงื่อนไขเพื่อตรวจสอบและรับเนื้อหาไฟล์ใบรับรองผู้ใช้

ขั้นตอนสุดท้ายคือฟังก์ชันถอดรหัสไฟล์ใบรับรองทั้งสองใบด้วย session key ดังแสดงใน ภาพที่ 67

```
//To decrypt Cert.
decrypt( mFileCAEncrpted, mFileCA );
decrypt( mFileClientEncrypted, mFileClient );

//To remove encrypted certs.
mFileCAEncrpted.delete();
mFileClientEncrypted.delete();
mCheck = true;
notificationAlert();
```

ภาพที่ 67 เมท็อดใช้ถอดรหัสไฟล์ใบรับรอง CA และใบรับรองผู้ใช้

ภาพที่ 68 คือแฟ้มที่เก็บไฟล์ใบรับรอง CA และ ใบรับรองตัวผู้ใช้ที่ถูกถอดรหัสและพร้อมใช้ติดตั้ง ซึ่งหากเนื้อหาข้อมูลใบรับรองที่ได้รับมีความถูกต้อง โปรแกรมจะสามารถทำการติดตั้งใบรับรองให้อัตโนมัติ

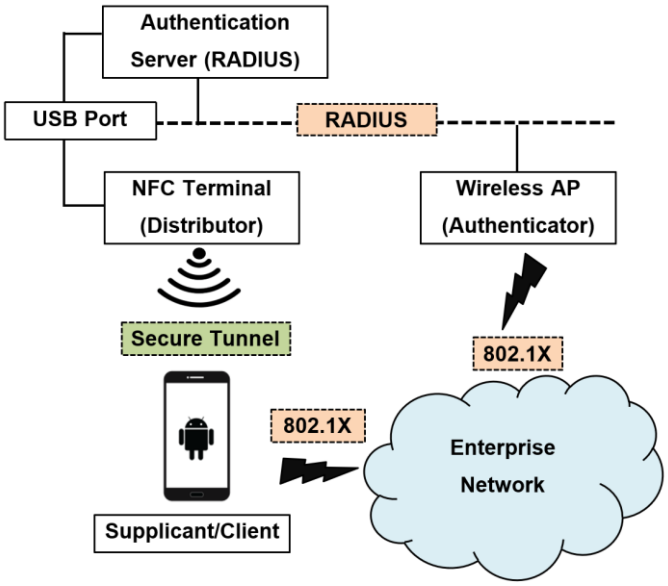


ภาพที่ 68 ไฟล์ใบรับรองที่ได้รับการถอดรหัสและพร้อมติดตั้ง

4.5 พัฒนาและเพิ่มฟังก์ชัน HMAC ให้กับการแลกเปลี่ยนข้อมูลผ่านเอ็นเอฟซี

4.5.1 สถาปัตยกรรมระบบ

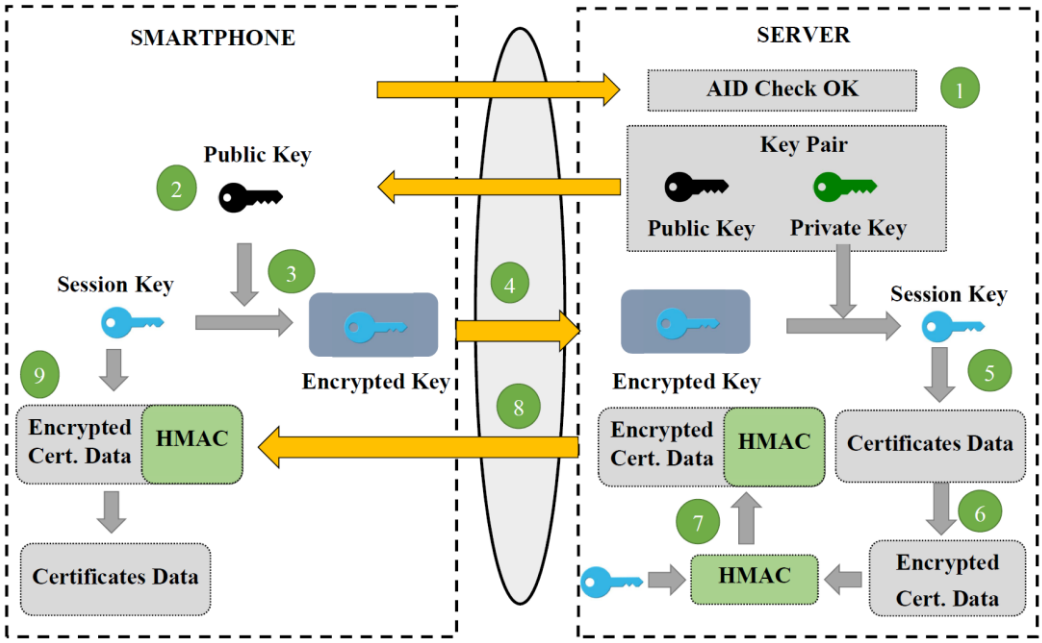
จากแนวทางเดิมที่เสนอไปนั้นผู้วิจัยเล็งเห็นว่าในกระบวนการแลกเปลี่ยนข้อมูลระหว่างอุปกรณ์สมาร์ตโฟนและตัวอ่านเอ็นเอฟซีนั้นยังสามารถเพิ่มความปลอดภัยและความถูกต้องของเนื้อหาข้อมูลที่ถูกขนส่ง ถึงแม้ว่าเอ็นเอฟซีจะมีข้อดีสำหรับความปลอดภัยในทางกายภาพ กล่าวคือผู้ใช้ต้องแตะสมาร์ตโฟนในระยะที่ชิฟเอ็นเอฟซีนั้นสามารถทำงานได้ซึ่งค่อนข้างใกล้กับตัวอ่านมาก แต่อย่างไรก็ตามเนื้อหาข้อมูลระหว่างการแลกเปลี่ยนนั้นยังคงสามารถถูกดักฟังและแก้ไขได้ โดยสถาปัตยกรรมที่พัฒนาเพิ่มเติมนี้ยังคงเหมือนเดิมดังแสดงในภาพที่ 69 ในส่วนที่เพิ่มเข้ามาคือฟังก์ชัน Hash-based Message Authentication Code (HMAC) ซึ่งสามารถใช้ตรวจสอบข้อความที่ได้รับนั้นถูกเปลี่ยนแปลงแก้ไขหรือไม่ โดยค่า HMAC ที่คำนวณได้จากข้อความจะถูกแทรกเข้าไปที่ส่วนท้ายของข้อความ



ภาพที่ 69 สถาปัตยกรรมชิ้นงาน

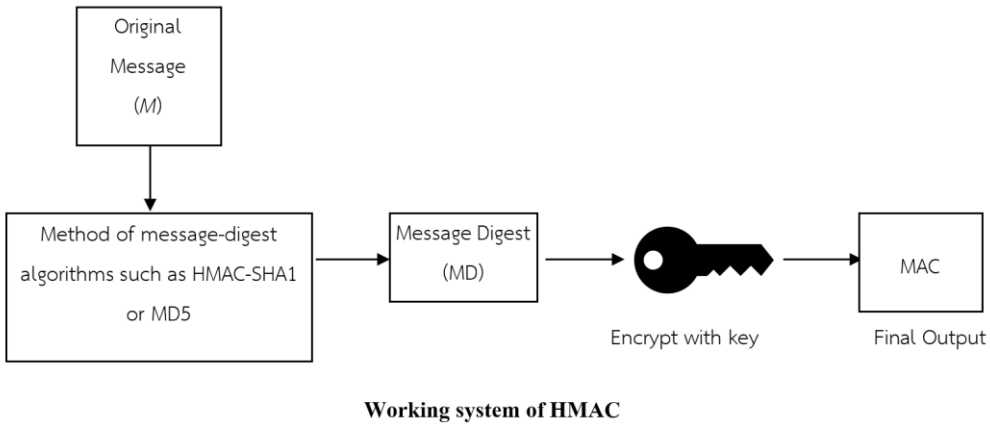
4.5.2 กระบวนการแลกเปลี่ยนข้อมูล

ภาพที่ 70 แสดงถึงกระบวนการทำงานในขั้นตอนการแลกเปลี่ยนข้อมูลใบรับรองระหว่างสมาร์โฟนและเครื่องบริการ โดยมีองค์ประกอบ HMAC หรือ ค่าแฮชเพิ่มเข้าไปในส่วนท้ายของเนื้อหาไฟล์ใบรับรอง เพื่อใช้ตรวจสอบและยืนยันความถูกต้องของเนื้อหาใบรับรอง



ภาพที่ 70 เครื่องบริการส่งข้อความเพื่อขอข้อมูลผู้ใช้

4.5.3 พัฒนาฟังก์ชัน



ภาพที่ 71 หลักการทำงานของ HMAC

เมท็อด genKeyHmac ทำหน้าที่คำนวณหาค่าแฮชจากไฟล์ใบรับรองซึ่งกำหนดให้ใช้อัลกอริทึม “HMACSHA1” ในการหาค่าแฮชดังแสดงในภาพที่ 72 โดยคีย์ลับที่ใช้คือ session key ที่ฝั่งเครื่องบริการได้รับจากสมาร์ตโฟน หลังจากกำหนดค่าเสร็จสมบูรณ์แล้ว เมท็อด doFinal ทำการคำนวณและให้ค่าคืนกลับเป็นอาร์เรย์ไบต์ เมื่อฝั่งเครื่องบริการทำการส่งข้อมูลใบรับรอง ค่าแฮชที่

คำนวณได้จะถูกนำมาต่อท้ายไฟล์ใบรับรองที่ถูกเข้ารหัส ค่าแฮชนี้จะถูกใช้ตรวจสอบก่อนทำการบันทึกและติดตั้งใบรับรองที่ฝั่งสมาร์ตโฟน

```
//Gen Key HMAC Function
protected byte[] genKeyHmac(byte[] msg) throws NoSuchAlgorithmException, InvalidKeyException, IOException {
    byte[] hmac = {};
    byte[] key = Files.readAllBytes(Paths.get(OUT_AES));
    SecretKey sKey = new SecretKeySpec(key, "AES");

    //Gen Message Authentication Code (MAC) for a message.
    Mac mac = Mac.getInstance(HMAC_SHA1_ALGORITHM);
    mac.init(sKey);
    try {
        hmac = mac.doFinal(msg);
        System.out.println("HMAC output: " + hmac);
        System.out.println("HMAC length: " + hmac.length);
        printByteArray(hmac);
    }catch (Exception e){
        System.out.println("HMAC Function Error is: " + e);
    }
    return hmac;
}
```

ภาพที่ 72 เมท็อด genKeyHmac ทำหน้าที่คำนวณหาค่าแฮชจากไฟล์ใบรับรอง

เมท็อด assertHmac ที่ฝั่งโปรแกรมเครื่องบริการทำหน้าที่ตรวจสอบค่าแฮชเพื่อยืนยันความถูกต้องก่อนแลกเปลี่ยนข้อมูลในขั้นตอนถัดไป ดังภาพที่ 73

```
protected boolean assertHmac(byte[] msg, byte[] hmac, byte[] chkHmac) throws InvalidKeyException,
                                                                    NoSuchAlgorithmException, IOException {
    hmac = genKeyHmac(msg);
    if(hmac.equals(chkHmac)) {
        return true;
    }else {
        return false;
    }
}
```

ภาพที่ 73 เมท็อด assertHmac ทำหน้าที่ตรวจสอบค่าแฮชจากข้อความ

สำหรับเมท็อด HMAC ที่ฝั่งสมาร์ตโฟนจะใช้เมท็อดเดียวกันกับฝั่งโปรแกรมเครื่องบริการเพื่อทำหน้าที่ตรวจสอบค่าแฮชเพื่อยืนยันความถูกต้องของเนื้อหาไฟล์เพื่อให้แน่ใจว่าไม่มีการเปลี่ยนแปลงหรือแก้ไขเกิดขึ้นระหว่างการขนส่ง

บทที่ 5
การทดสอบและการวิเคราะห์ผล

5.1 วัตถุประสงค์ของการทดสอบ

เพื่อทดสอบประสิทธิภาพในใช้งานจริงกับระบบการพิสูจน์ตัวจริงแบบหลายปัจจัยผ่านเอ็นเอฟซีบนสามาร์ทโฟน โดยการทดสอบเพื่อประเมินและวัดผลแบ่งออกเป็นสองส่วนหลัก ได้แก่

1) การประเมินค่าเวลาที่ใช้ในระบบ ประกอบไปด้วย 1) เวลาที่ใช้ในการแลกเปลี่ยนใบรับรอง และ 2) เวลาที่ใช้ในการพิสูจน์ตัวจริงจนเชื่อมต่อเครือข่ายไร้สายสำเร็จ โดยทำการทดลอง 100 ครั้งที่ความมั่นใจ 95% ผลสุดท้ายจากการวิเคราะห์ในแต่ละส่วนคือ ค่าเวลามากสุด ค่าเวลาน้อยสุด ค่าเวลาเฉลี่ย และค่าเวลาเฉลี่ยรวม

2) การประเมินประสบการณ์ผู้ใช้ โดยออกแบบแบบสำรวจความพึงพอใจจากการทดลองใช้งานจริงจากพนักงานในบริษัท ประกอบไปด้วยการประเมินใน 5 มิติคุณภาพ (Quality Dimension)

1. อรรถประโยชน์ (Utility) การทำงานของแอปพลิเคชันสามารถเข้าใจง่ายและทำงานได้เหมาะสมตามวัตถุประสงค์ด้วยการใช้ใบรับรองเพื่อเชื่อมต่อ Wi-Fi ในองค์กร
2. การใช้งาน (Usability) แอปพลิเคชันสามารถใช้งานง่ายและมีประสิทธิภาพเมื่อเปรียบวิธีการรับใบรับรองและติดตั้งใบรับรองแบบอื่น ๆ ในปัจจุบัน เช่น ผ่านอีเมล หรือเว็บไซต์ที่ปลอดภัย
3. ความถูกต้อง (Correctness) ผลลัพธ์ที่ได้มีความถูกต้องตามที่คาดหวังแก่การนำมาใช้งานจริงได้
4. คุณค่า (Value) แอปพลิเคชันมีคุณค่าและควรนำมาใช้งานในทางปฏิบัติกับองค์กร
5. ความพึงพอใจโดยรวม (Overall Satisfaction) ความพึงพอใจโดยรวมต่อแอปพลิเคชันและระบบการพิสูจน์ตัวจริงที่นำเสนอ

5.1.1 การประเมินค่าเวลาที่ใช้ในระบบ

การทดสอบจะทำการจับเวลาในแต่ละส่วน ก) การแลกเปลี่ยนข้อมูลใบรับรองและข้อมูลตั้งค่าเครือข่าย ดังแสดงในภาพที่ 74 และ ข) การพิสูจน์ตัวจริงกับ AP จนถึงสถานะเชื่อมต่อเครือข่ายไร้สายสำเร็จดังแสดงในภาพที่ 75 ทั้งนี้ไม่นับรวมเวลาที่ผู้ใช้กรอกรหัสผ่านและใช้ลายนิ้วมือเพื่อติดตั้งใบรับรอง โดยกำหนดตัวนับเวลาไว้ในโค้ดของแอปพลิเคชัน เมื่อแต่ละส่วนทำงานสำเร็จแอปพลิเคชันจะแสดงผลลัพธ์ค่าเวลาออกมาทางหน้าจอ จากนั้นทำการบันทึกค่าและทำซ้ำจนครบ 100 ครั้ง

```
//Notification
private void notificationAlert2()
    {
        tEnd = System.currentTimeMillis();
        tDelta = tEnd - tStart;
        elapsedSeconds = tDelta / 1000.0;
        Log.d( tag: "TIME", msg: "Time completed to exchange Certificates: " + elapsedSeconds + " Seconds");
        vibrate();
        Toast.makeText( context: this,  text: "Transfer Certificates Done ! ",Toast.LENGTH_SHORT).show();
    }
```

ภาพที่ 74 ตรวจสอบเวลาที่ใช้รับใบรับรองและข้อมูลตั้งค่าเครือข่าย

```
/**Delta time counter**/
 tDelta = tEnd - tStart;
 elapsedSeconds = tDelta / 1000.0;
 Log.d( tag: "TIME", msg: "Time completed to connect Wi-Fi: " + elapsedSeconds  + " Seconds");
```

ภาพที่ 75 ตรวจสอบเวลาที่ใช้พิสูจน์ตัวจริงและเชื่อมต่อ Wi-Fi

ขั้นตอนการทดลอง

1) ผู้ใช้แตะสมาร์ตโฟนที่ตัวอ่านเอ็นเอฟซีเพื่อเริ่มการแลกเปลี่ยนข้อมูล
2) หลังจากที่ได้รับข้อความแจ้งเตือน “Transfer Done !”ผู้ใช้นำเอาสมาร์ตโฟนออกจากตัวอ่านเอ็นเอฟซี
3) ทำการบันทึกผลเวลาที่ได้ลงใน Excel ชีท
4) ทำซ้ำจากขั้นที่ 1 ถึง 3 จนครบ 100 รอบ

ขั้นตอนการหาค่าความเชื่อมั่น

การทดลองเพื่อหาผลลัพธ์เวลาจากค่าความเชื่อมั่นนี้อ้างอิงจาก [2] โดยทำการทดลองและบันทึกผลในแต่ละรอบจนครบ 100 ครั้ง จากนั้นนำผลที่ได้มาคำนวณหาค่าเวลาซึ่งใช้ค่าความมั่นใจ (Confidence interval) ที่ 95% โดยใช้โปรแกรม Excel ช่วยในการคำนวณประกอบด้วยขั้นตอนดังนี้

1). คำนวณค่าเฉลี่ยและส่วนเบี่ยงเบนมาตรฐานจากค่าที่บันทึก 100 ครั้ง ของแต่ละส่วน

$$\bar{x} = \frac{\sum x}{n} \qquad \sigma = \sqrt{\frac{\sum(x - \bar{x})}{n}}$$

2). เลือกค่าความมั่นใจซึ่งในที่นี้เลือกที่ 95% และค่าวิกฤติที่ได้คือ 1.96 หาได้จากการเปิดตาราง Z-Table

3) หาค่าความคลาดเคลื่อน (Margin of Error)

$$z\alpha_{/2} \times \frac{\bar{x}}{\sqrt{(n)}}$$

4) บอกช่วงค่าความเชื่อมั่น

$$\bar{x} \pm z\alpha_{/2} \times \frac{\sigma}{\sqrt{(n)}}$$

โดยค่าที่ทั้งหมดหาได้จากขั้นตอน 1 ถึง 4 ข้างต้น แสดงดังภาพที่ 76

Iteration	Exchange data over NFC (S)	Connect Wi-Fi (S)	Total (S)
97	3.755	3.245	7
98	3.685	3.249	6.934
99	3.88	3.248	7.128
100	3.956	3.247	7.203
Mean	3.8560	3.2539	7.1099
Standard Deviation	0.340	0.018	0.343
Sample Size	100	100	100
Confidence Coff. (95%)	1.96	1.96	1.96
Margin of Error	0.06658	0.00355	0.06726
Upper Bound	3.92259	3.25743	7.17715
Lower Bound	3.78943	3.25033	7.04263
Max	5.02800	3.35200	8.29300
Min	3.65700	3.23500	6.90300
Range	1.37100	0.11700	1.39000

Composition	Max (s)	Min (s)	Average (s)
Obtain credentails over NFC	5.028	3.657	3.856 ± 0.0665
Connect Wi-Fi	3.352	3.235	3.254 ± 0.0035
Total	8.293	6.903	7.11 ± 0.0673

ภาพที่ 76 ผลลัพธ์ที่ได้จากการคำนวณ

ตารางที่ 2 แสดงให้เห็นถึงผลลัพธ์ของเวลาที่ได้จากการแลกเปลี่ยนใบรับรองผ่านเอ็นเอฟซีโดยไม่ใช้ฟังก์ชัน HAC ประกอบด้วย ค่าเวลาต่ำสุด ค่าเวลาเฉลี่ยต่ำสุด เวลาสูงสุด ค่าเวลาเฉลี่ยสูงสุด และค่าเวลาเฉลี่ยรวม

ตารางที่ 2 ผลลัพธ์เวลาช่วงค่าความเชื่อมั่นที่ 95% ของแต่ละส่วนแบบไม่มีฟังก์ชัน HMAC

Components	*Min (s)*	*Max (s)*	*Average (s)*
Exchange credentials data over NFC	3.657	5.028	3.856 ± 0.0665
Connect to Wi-Fi & Authentication	3.235	3.352	3.254 ± 0.0035
Total	6.903	8.293	7.110 ± 0.0673

โดยค่าที่ทั้งหมดหาได้จากขั้นตอน 1 ถึง 4 ข้างต้น แสดงดังภาพที่ 77

Iteration	Exchange data over NFC (S)	Connect Wi-Fi (S)	Total (S)
96	5.229	3.239	8.468
97	5.209	3.236	8.445
98	5.25	3.241	8.491
99	5.259	3.25	8.509
100	5.329	3.245	8.574
Mean	5.3039	3.2483	8.5522
Standard Deviation	0.118	0.012	0.119
Sample Size	100	100	100
Confidence Coff. (95%)	1.96	1.96	1.96
Margin of Error	0.02309	0.00228	0.02333
Upper Bound	5.32701	3.25060	8.57557
Lower Bound	5.28083	3.24604	8.52891
Max	5.96200	3.30400	9.22000
Min	5.18500	3.23200	8.42900
Range	0.77700	0.07200	0.79100

Composition	Max (s)	Min (s)	Average (s)
Exchange data over NFC	5.96200	5.18500	5.303 ± 0.023
Connect Wi-Fi	3.30400	3.23200	3.248 ± 0.002
Total	9.22000	8.42900	8.552 ± 0.023

ภาพที่ 77 ผลลัพธ์ที่ได้จากการคำนวณ

ตารางที่ 3 แสดงให้เห็นถึงผลลัพธ์ของเวลาที่ได้จากการแลกเปลี่ยนใบรับรองผ่านเอ็นเอฟซีด้วยร่วมกับฟังก์ชัน HMAC โดยประกอบด้วย ค่าเวลาต่ำสุด ค่าเวลาเฉลี่ยต่ำสุด เวลาสูงสุด ค่าเวลาเฉลี่ยสูงสุด และค่าเวลาเฉลี่ยรวม

ตารางที่ 3 ผลลัพธ์เวลาช่วงค่าความเชื่อมั่นที่ 95% ของแต่ละส่วนเมื่อเพิ่มฟังก์ชัน HMAC

Components	*Min (s)*	*Max (s)*	*Average (s)*
Exchange credentials data over NFC	5.185	5.962	5.303 ± 0.023
Connect to Wi-Fi & Authentication	3.323	3.304	3.248 ± 0.002
Total	8.429	9.22	8.552 ± 0.023

5.1.2 การประเมินประสบการณ์ผู้ใช้

ในส่วนของการประเมินประสบการณ์ผู้ใช้นั้น การเลือกกลุ่มตัวอย่างการประเมินจะเลือกเฉพาะกลุ่มพนักงานที่ทำงานในบริษัทไอทีซึ่งบริษัทที่ทำงานนั้นมีการให้บริการเครือข่ายไร้สายให้กับพนักงาน โดยการประเมินจะจัดในลักษณะของกลุ่มที่สนใจ 17 คน ทั้งนี้แบบประเมินความพึงพอใจแสดงในภาคผนวก ก โดยระดับการประเมินความพึงพอใจแบ่งเป็น 5 ระดับ

1 ไม่พึงพอใจอย่างมาก

2 ไม่พึงพอใจ

3 ปานกลาง

4 พึงพอใจ

5 พึงพอใจอย่างมาก

ผู้ทดลองใช้แอปพลิเคชันมีทั้งหมด 17 คน ซึ่งเป็นพนักงานของบริษัทไอทีแห่งหนึ่งซึ่งสนับสุนนโยบาย BYOD โดยอนุญาตให้พนักงานสามารถนำอุปกรณ์ของตนเองมาเชื่อมต่อเครือข่ายไร้สายที่จัดเตรียมไว้ได้

ตารางที่ 4 ผลการประเมินประสบการณ์ผู้ใช้

	คะแนน					คะแนนเฉลี่ย
อรรถประโยชน์ (Utility) การทำงานของแอปพลิเคชันสามารถเข้าใจง่ายและทำงานได้เหมาะสมตามวัตถุประสงค์	5	4	3	2	1	
	0	5	10	2	0	3.17
การใช้งาน (Usability) แอปพลิเคชันสามารถใช้งานง่ายและมีประสิทธิภาพเมื่อเปรียบวิธีการรับใบรับรองและติดตั้งใบรับรองแบบอื่น ๆ	0	7	8	2	0	3.30
ความถูกต้อง (Correctness) ผลลัพธ์ที่ได้มีความถูกต้องตามที่คาดหวังแก่การนำมาใช้งานจริงได้	17	0	0	0	0	5
คุณค่า (Value) แอปพลิเคชันมีคุณค่าและควรนำมาใช้งานในทางปฏิบัติกับองค์กร	0	6	9	2	0	3.23
ความพึงพอใจโดยรวม (Overall Satisfaction) ความพึงพอใจโดยรวมต่อแอปพลิเคชันและระบบการพิสูจน์ตัวจริงที่นำเสนอ	0	7	10	0	0	3.41
ค่าเฉลี่ยโดยภาพรวม						3.62

บทที่ 6
สรุปผลการวิจัย

6.1 สรุปผลการวิจัย

การใช้วิธีการพิสูจน์ตัวจริงด้วยใบรับรองถือเป็นวิธีการที่มีความปลอดภัยสูงแต่ข้อเสียเปรียบประการหนึ่งคือความยากในการจัดการใบรับรอง งานวิจัยนี้ใช้ประโยชน์จากชิปเอ็นเอฟซีที่มีอยู่ในสมาร์ตโฟนเป็นช่องทางในการแลกเปลี่ยนข้อมูลใบรับรองซึ่งสามารถช่วยลดภาระของผู้ดูแลระบบและผู้ใช้งานได้ ในขณะเดียวกันการพิสูจน์ตัวจริงแบบหลายปัจจัยถูกนำมาใช้เพื่อช่วยให้แน่ใจว่าแต่ละขั้นตอนจะต้องได้รับการยืนยันจากผู้ใช้ก่อน เพื่อเป็นการเพิ่มความแข็งแรงให้กับระบบความปลอดภัยอีกระดับ จากผลการทดลองพบว่าเวลาที่ใช้ในการขอใบรับรองและการเข้าถึงอินเทอร์เน็ตสามารถทำได้ภายในเวลาไม่เกิน 10 วินาทีซึ่งถือว่ารวดเร็วและจัดการง่ายเมื่อเปรียบเทียบกับวิธีการติดตั้งใบรับรองอื่นๆ งานวิจัยนี้ยังทำการประเมินประสบการณ์ผู้ใช้จำนวน 17 คนซึ่งเป็นพนักงานไอที ได้ลงความเห็นว่าผลลัพธ์ที่คาดหวังของแอปพลิเคชันในการเชื่อมต่อเครือข่ายไร้สายสามารถทำงานได้อย่างถูกต้อง และความเหมาะสมแก่การนำมาใช้งานกับระบบการพิสูจน์ตัว จริงของเครือข่ายไร้สายในองค์กรที่ระดับปานกลางจากผลตารางที่ 4 เนื่องจากยังมีข้อจำกัดของอุปกรณ์ที่สนับสนุนเทคโนโลยีเอ็นเอฟซี ได้แก่ อุปกรณ์ Apple ที่อนุญาตให้ใช้งานได้กับบางฟังก์ชันเท่านั้น เช่น บริการจ่ายเงินกับ Apple pay หรือซิงค์ข้อมูลกับ Apple Watch นอกจากนี้ความกังวลในด้านความปลอดภัยของตัวผู้ใช้งานต่อเทคโนโลยีนี้ยังมีผลต่อการเลือกใช้งานเอ็นเอฟซีเพื่อเป็นทางเลือกในการถ่ายโอนข้อมูลที่สำคัญ

6.2 ข้อจำกัดในงานวิจัย

6.2.1 ในส่วนของอุปกรณ์ที่สนับสนุนเอ็นเอฟซียังคงมีจำนวนไม่มาก ทำให้การนำไปใช้งานจริงนั้นค่อนจำกัดเฉพาะอุปกรณ์ เช่น สมาร์ตโฟน หรือ แทบเล็ต บางกลุ่มเท่านั้น

6.2.2 ความปลอดภัยบนระบบปฏิบัติการแอนดรอยด์ยังคงไม่ปลอดภัย หากผู้ใช้มีสิทธิ์เป็น rooted แล้ว ข้อมูลสำคัญสามารถถูกดึงออกมาจากแฟ้มและแก้ไขได้

6.2.3 แอปพลิเคชันมีความเป็นไปได้ที่จะถูก decompile เพื่อหา logic ของโปรแกรม

6.3 งานวิจัยในอนาคต

งานในอนาคตที่ควรศึกษาเพิ่มเติมคือการนำเอ็นเอฟซีไปใช้แลกเปลี่ยนข้อมูลความลับและประยุกต์ใช้งานกับทุกวิธีมาตรฐานการพิสูจน์ตัวจริงของเครือข่ายไร้สาย

รายการอ้างอิง

1. K. Hajdarevic, P.A., M. Spremic, *Proactive security metrics for Bring Your Own Device (BYOD) in ISO 27001 supported environments*, in *2016 24th Telecommunications Forum (TELFOR)*.
2. A. Matos, D.R., P. Trezentos, *Secure hotspot authentication through a Near Field Communication side-channel*, in *2012 IEEE 8th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. 2012.
3. Kerner, S.M. *EAP-TLS Detailed as WiFi Security Best Practice at SecTor*. 2017 01-Dec-2017]; Available from: http://www.eweek.com/security/eap-tls-detailed-as-wifi-security-best-practice-at-sector.
4. Simon D., A.B., Hurst R. , *The EAP-TLS Authentication Protocol*. 2008.
5. Radivilova, T., H. A. Hassan, *Test for penetration in Wi-Fi network: Attacks on WPA2-PSK and WPA2-enterprise*, in *2017 International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo)*. 2017.
6. Vedat Coskun, K.O., Busra Ozdenizci, *Near Field Communication: From Theory to Practice*. 2011, 27 December 2011.
7. Van der Walt, D., *FreeRADIUS: beginner's guide ; manage your network resources with FreeRADIUS*. Birmingham: Packt Publ.
8. Choudhury, S., Bhatnagar, Kartik, Haque, Wasim, *Public Key Infrastructure Implementation and Design.*
9. C. Sudar, S.K.A., L. R. Deepthi, *Time-based one-time password for Wi-Fi authentication and security*, in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. 2017.
10. Purevjav, S., Kim, T., Lee, H., *Email encryption using hybrid cryptosystem based on Android*, in *2016 18th International Conference on Advanced Communication Technology (ICACT)*.

ภาคผนวก

ภาคผนวก ก

User Experience Assessments (UXA)

จุดพร้อมโยงที่ปลอดภัยด้วยการพิสูจน์ตัวจริงแบบหลายปัจจัยผ่านเอ็นเอฟซีบนสมาร์ตโฟน

(**Secure Hotspots with Multi-Factor Authentication Through NFC on Smartphones**)

……/ ……/…… (DD/ MM/ YYYY)

Objective: To survey and evaluate the quality of authentication system for secure hotspots with multi-factor authentication through **NFC** on smartphones. Please kindly help us to fill in the information and mark √ in the level of satisfaction that best suits your feelings and experiences.

Explanation: All personal information is kept as confidentiality. The assessment consists of 3 sections as follows.

Section 1 General user information

Section 2 Explore user experience

Section 3 User satisfaction assessment

Section: 1 General User Information	
Name…… **Company**……	
Nationality	☐ Thai ☐ German ☐ Indonesian ☐ Indian ☐ Philippines ☐ Please specify……
Work experience	☐ 1-5 Years ☐ 6-10 Years ☐ > 10 Years
Gender	☐ Male ☐ Female
Works Position	☐ SAP ABAP Consultant ☐ SAP Functional Consultant ☐ Network Admin ☐ SAP Basis Consultant ☐ Etc.Please specify……
Section: 2 Explore User Experiences	
1. Have you ever brought your smartphone to connect the corporate network? ☐ Yes ☐ No Reason……	
2. Do you think that the security level of your corporate wireless network is secure? ☐ Yes ☐ No Reason……	

3. What is the purpose(s) of using your smartphone to connect the corporate Wi-Fi network? (more than 1 answer can be chosen)

☐ Support work ☐ Communication ☐ Browsing information

☐ Please specify……

4. Have you ever used certificates to authenticate for corporate Wi-Fi?

☐ Yes ☐ No Reason……

5. How do you think about the complexity of corporate Wi-Fi authentication process on smartphones?

☐ Easy ☐ Average ☐ Hard ☐ Super Hard

Reason……

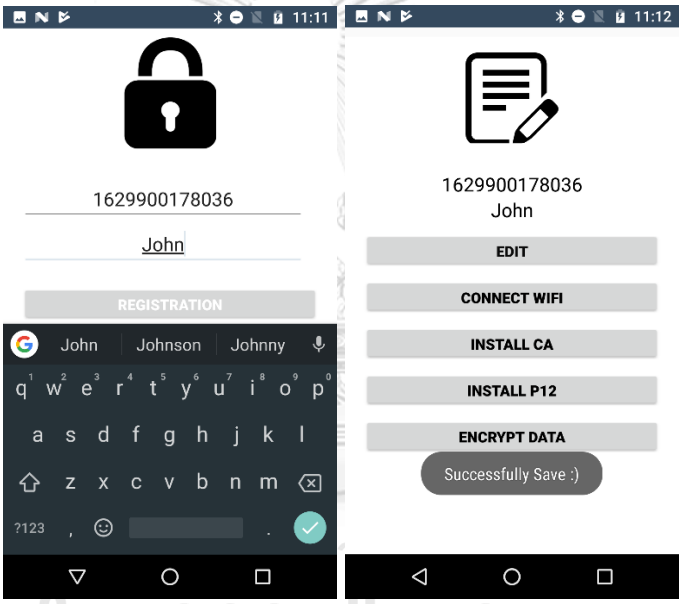
Section: 3 User Satisfaction Assessment

User Experience Aspects	**Satisfaction Level**				
	Very Satisfied (5)	Satisfied (4)	Neutral (3)	Unsatisfied (2)	Very Unsatisfied (1)
1. The application functionality is easy to comprehend and achieves certificate purpose for corporate Wi-Fi access					
2. The application is user-friendly and more efficient than other certificate installation methods such as encrypted email or secure certificate store					
3. The outcome is accurate as expected in practical					
4. The application is valuable and appropriate in practical usage for corporate Wi-Fi					
5. Overall satisfaction with the application and authentication system					

ภาคผนวก ข

คู่มือการใช้งาน

สำหรับวิธีการใช้งานแอปพลิเคชันเพื่อใช้รับใบรับรองและพิสูจน์ตัวจริงเพื่อเชื่อมต่อเครือข่ายไร้สายในองค์กร ผู้ใช้จะต้องทำการลงทะเบียนโดยกรอกรหัสประจำตัวผู้ใช้และชื่อ จากนั้นกดปุ่มลงทะเบียน (Registration) ดังภาพที่ 78 (ก) ซึ่งจะมีข้อความแจ้งเตือนผู้ใช้ให้ทราบถึงสถานะการบันทึกไฟล์ข้อมูลสำเร็จ ดังภาพที่ 78 (ข) เมื่อผู้ใช้ทำการลงทะเบียนสำเร็จ ข้อมูลส่วนตัวผู้ใช้จะแสดงบนหน้าจอ ซึ่งยังเป็นข้อมูลเปล่า (plain text) โดยจะถูกเข้ารหัสด้วยกุญแจสาธารณะของเครื่องบริการภายหลังเมื่อผู้ใช้แตะสมาร์ตโฟนกับตัวอ่านเอ็นเอฟซี



(ก) (ข)

ภาพที่ 78 (ก) ผู้ใช้กรอกรหัสประจำตัวผู้ใช้และชื่อ และ (ข) ข้อความแจ้งสถานะ

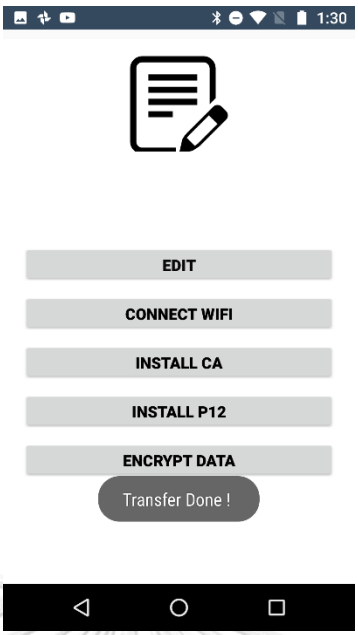
ขั้นตอนถัดไปคือผู้ใช้แตะสมาร์ตโฟนกับตัวอ่านเอ็นเอฟซีเพื่อแลกเปลี่ยนข้อมูลระหว่างสมาร์ตโฟนและเอ็นเอฟซีเดสก์ท็อปโปรแกรมดังภาพที่ 79 (ก) แสดงถึงสถานะตัวอ่านเอ็นเอฟที่พร้อมใช้งาน และภาพที่ 79 (ข) ผู้ใช้แตะสมาร์ตโฟนกับตัวอ่านในระยะไม่เกิน 5 เซนติเมตร เพื่อเริ่มดำเนินการรับส่งข้อมูล โดยในระหว่างการแตะนั้นผู้ใช้จะต้องรอจนกว่าการขนส่งข้อมูลจะเสร็จสมบูรณ์โดยจะมี pop-up พร้อมการสั่นเพื่อแจ้งเตือนผู้ใช้ดังภาพที่ 80 ซึ่งหากการแตะถูกยกเลิกหรือ

ผู้ใช้ปิดแอปพลิเคชันอาจทำให้การส่งข้อมูลไม่สำเร็จทำให้ข้อมูลใบรับรองและข้อมูลการตั้งค่าเครือข่ายเกิดข้อความผิดพลาด

(ก) (ข)

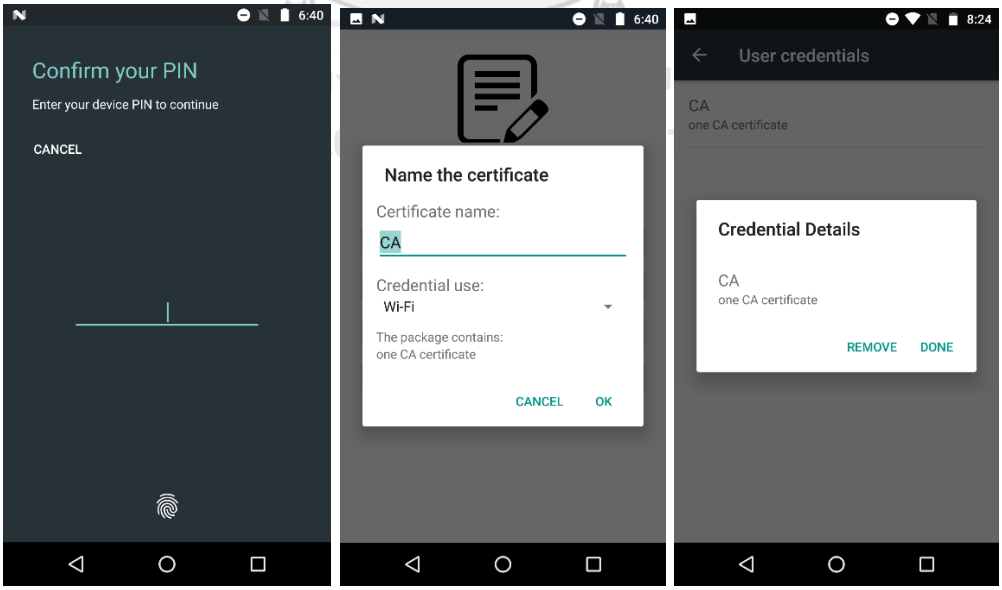
ภาพที่ 79 (ก) สถานะของตัวอ่านพร้อมใช้งาน และ (ข) ผู้ใช้แตะสมาร์ตโฟนกับตัวอ่าน

ในระหว่างการแลกเปลี่ยนข้อมูลอุปกรณ์ทุกเครื่องที่แตะกับตัวอ่านเอ็นเอฟซีจะได้รับกุญแจสาธารณะของเครื่องบริการเพื่อใช้เข้ารหัสข้อมูลส่วนตัวและใช้เพื่อส่ง session key ที่ใช้ในการเข้ารหัสไฟล์ใบรับรอง เมื่อการขนส่งใบรับรองสำเร็จจะมีข้อความแจ้งผู้ใช้พร้อมการสั่นและข้อมูลส่วนตัวผู้ใช้ได้หายไปเนื่องจากถูกเข้ารหัสแล้ว

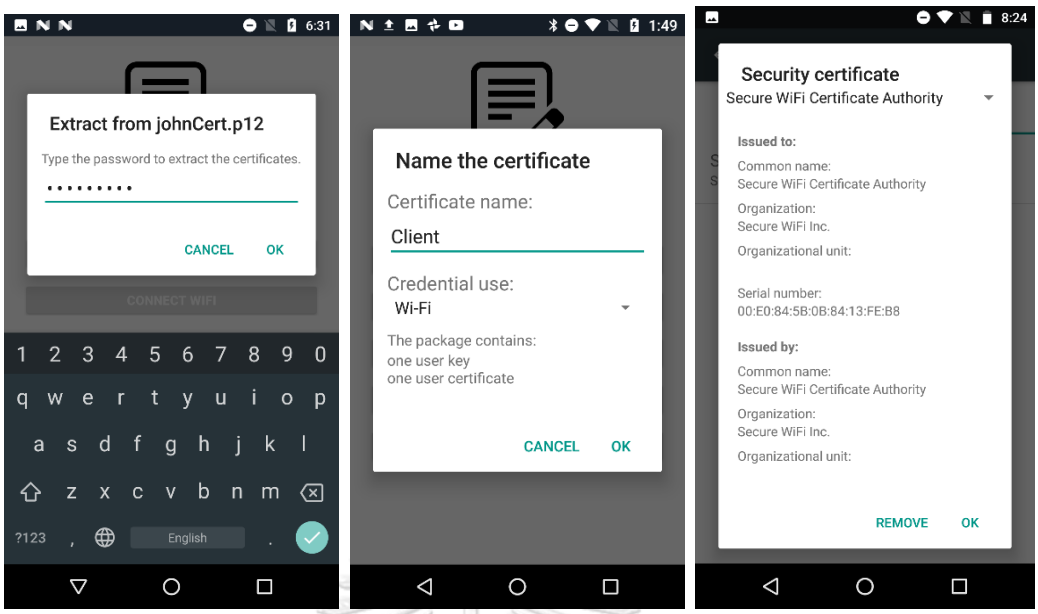


ภาพที่ 80 สถานะการขนส่งข้อมูลเสร็จสมบูรณ์

เมื่อฝั่งอุปกรณ์ของผู้ใช้ได้รับใบข้อมูลรับรองและข้อมูลการตั้งค่าเครือข่ายครบสมบูรณ์ แอปพลิเคชันจะเรียกฟังก์ชันผู้ช่วยติดตั้งใบรับรอง โดยเริ่มจากใบรับรอง root CA ซึ่งในขั้นตอนนี้ผู้ใช้ต้องยืนยันด้วยลายนิ้วมือหรือ PIN ก่อนดังภาพที่ 81 และหลังจากนั้นใบรับรองที่ลงต่อมาคือใบรับรองผู้ใช้ดังภาพที่ 82

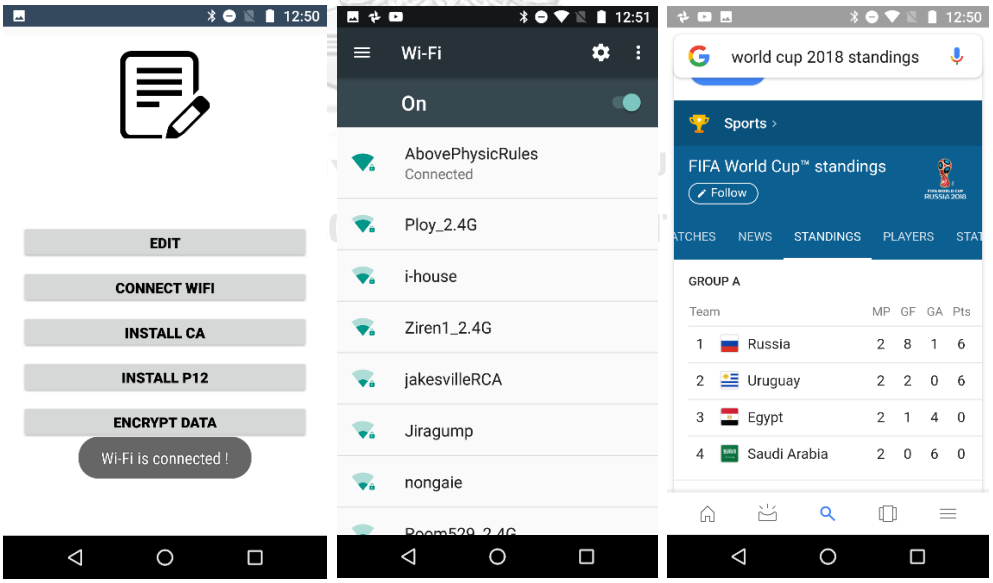


ภาพที่ 81 รายละเอียดการติดตั้งใบรับรอง root CA



ภาพที่ 82 รายละเอียดการติดตั้งใบรับรองผู้ใช้

หลังจากการนำเข้าและติดตั้งใบรับรองสำเร็จ แอปพลิเคชันจะเรียกฟังก์ชันผู้ช่วยในการเชื่อมต่อเครือข่ายเป้าหมายอัตโนมัติ เมื่อสถานะการเชื่อมต่อสำเร็จจะแสดงข้อความแจ้งเตือนผู้ใช้ โดยรายละเอียดแสดงดังภาพที่ 83



ภาพที่ 83 สถานะแอปพลิเคชันเมื่อเชื่อมต่อเครือข่ายไร้สายสำเร็จ

ประวัติผู้เขียนวิทยานิพนธ์

นาย วิภพ โพธิ์มาก เกิดเมื่อวันที่ 16 กุมภาพันธ์ พ.ศ. 2533 ที่จังหวัดกำแพงเพชร สำเร็จการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต (วศ.บ.) สาขาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี ในปีการศึกษา 2555 และเข้าศึกษาต่อในหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิทยาศาสตร์คอมพิวเตอร์ ที่ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2559