เทคนิคการพิสูจน์ตัวจริงภายในโดเมนอย่างปลอดภัยและเบาสำหรับโพรโตคอล SIP

กศิณ นุตาลัย[1] และ สุภกร กังพิศดาร[2]

คณะวิทยาการและเทคโนโลยีสารสนเทศ มหาวิทยาลัยเทคโนโลยีมหานคร

Emails:[1] kasin1985@gmail.com, [2]supakorn@mut.ac.th

บทคัดย่อ

Session Initiation Protocol (หรือ SIP) ได้ถูกนำมาใช้อย่างแพร่หลายในการจัดการการเชื่อมต่อของ Voice Over IP อย่างไรก็ตามมาตรฐานของการเชื่อมต่อของ SIP นั้นยังไม่มีคุณสมบัติทางด้านความมั่นคงปลอดภัยที่ดีโดยเฉพาะอย่างยิ่งคุณสมบัติการยืนยันตัวตนของผู้ใช้งาน (User Authentication) แม้ว่างานวิจัยต่างๆ ได้ถูกเสนอขึ้นเพื่อเพิ่มความมั่นคงปลอดภัยให้แก่โพรโตคอล SIP แต่ก็ยังขาดคุณสมบัติที่จำเป็น งานวิจัยฉบับนี้เสนอแนวทางการเพิ่มความสามารถในการยืนยันตัวตนของผู้ใช้งานโพรโตคอล SIP ภายใต้โดเมนเดียวกัน ด้วยการนำเอาเทคนิคการเข้ารหัสลับแบบสมมาตรและการกระจายเซสชั่นคีย์แบบออฟไลน์ทำให้การยืนยันตัวตนของผู้ใช้งานทำได้อย่างมีความมั่นคงปลอดภัยและลดเวลาในการทำงานลงเมื่อเปรียบเทียบกับวิธีการที่มีอยู่ นอกจากนี้ยังเพิ่มคุณสมบัติการยืนยันตัวตนของเซิร์ฟเวอร์ SIP ต่อผู้ใช้งานอีกด้วย

คำสำคัญ-- เซสชั่นคีย์แบบออฟไลน์; การยืนยันตัวตน; ค่าที่มีการรับรอง; วิธีการแลกเปลี่ยนคีย์แบบ DH half-key

1. บทนำ

ในปัจจุบัน การใช้งาน VOIP (Voice Over Internet Protocol) ได้รับความนิยมอย่างแพร่หลาย เนื่องจากสามารถลดค่าใช้จ่ายของการใช้โทรศัพท์และทำให้การติดต่อสื่อสารได้สะดวกยิ่งขึ้น ดังนั้นจึงได้มีการพัฒนาเทคโนโลยี VoIP อย่างต่อเนื่อง หนึ่งในเทคโนโลยีการสื่อสารชนิด VoIP คือโพรโตคอล Session Initiation Protocol (หรือ SIP) โพรโตคอล SIP ทำหน้าที่จัดการการเชื่อมต่อระหว่างผู้ใช้ (User Agent หรือ UA) เช่น ทำหน้าที่กำหนดลักษณะการตกลงร่วมกันว่าจะมีการส่งข้อมูลอย่างไร การเลือกใช้ Codec เป็นต้น

ถึงแม้ว่าโพรโตคอล SIP จะได้รับความนิยมเพิ่มขึ้น แต่อย่างไรก็ตามโพรโตคอลนี้ก็ยังมีข้อจำกัดทางด้านความมั่นคงปลอดภัยค่อนข้างมาก ไม่ว่าจะเป็นในส่วนของการระบุตัวตน, การตรวจสอบความถูกต้องของข้อมูล, การส่งข้อมูลโดยไม่ถูกการดักจับ (Sniff) ทั้งหมดข้างต้นนั้นเนื่องมาจากโพรโตคอล SIP มีพื้นฐานมาจากโพรโตคอล HTTP (Hypertext Transfer Protocol) ซึ่งใช้การส่งข้อมูลในลักษณะ plaintext เป็นหลัก ซึ่งตาม RFC 3261 [2] นั้นพบว่าโพรโตคอล SIP ยังไม่มีการระบุระบบความมั่นคงปลอดภัยที่ชัดเจน โดยเฉพาะอย่างยิ่งในระดับชั้น (Layer) ของโพรโตคอล SIP เอง โดยส่วนมากแล้วความมั่นคงปลอดภัยของโพรโตคอล SIP มักเป็นการใช้เทคโนโลยีความมั่นคงปลอดภัยจากระดับชั้นอื่นเป็นหลัก เช่น มีการใช้โพรโตคอล TLS (Transport Layer Security) เพื่อความมั่นคงปลอดภัยของการสื่อสารระหว่างผู้ใช้ (UA) และพร็อกซี่เซิร์ฟเวอร์ (Proxy Server หรือ Server) การเลือกใช้เทคโนโลยีการรักษาความมั่นคงปลอดภัยจากระดับชั้นอื่นทำให้เกิดการคำนวณรวมทั้งปริมาณ Overheadในระหว่างการสร้างเซสชั่น (Session) ที่เพิ่มขึ้นในกรณีที่แบนด์วิท (Bandwidth) ถูกจำกัด นอกจากนี้โพรโตคอล SIP ยังมีช่องโหว่อื่นๆ อีก เช่น การยืนยันตัวตนของผู้ใช้ (User Authentication) นั้นสามารถทำได้แค่เพียงฝั่งเดียว คือ เป็นการยืนยันตัวตนของ UA ต่อ SIP Server (หรือ Server) เท่านั้น นอกจากนี้ระดับความมั่นคงปลอดภัยในการส่งข้อมูลระหว่างกันยังมีข้อจำกัดอยู่

งานวิจัยจำนวนมากได้ถูกเสนอขึ้นเพื่อเพิ่มความมั่นคงปลอดภัยในการส่งข้อมูลสำหรับโพรโตคอล SIP [3, 6] หนึ่งในงานวิจัยที่น่าสนใจได้แก่ งานของ Choi *et al.* [3] ได้เสนอโพรโตคอลสำหรับการยืนยันตัวตนของผู้ใช้และมีคุณสมบัติความมั่นคงปลอดภัยระหว่าง Hop เพื่อนำมาแทนที่การใช้โพรโตคอล TLS ในการเข้ารหัสลับข้อมูล นอกจากนี้ยังมีการเพิ่มการแลกเปลี่ยนเซสชั่นคีย์แบบ DH half-key จะช่วยให้มีความมั่นคงปลอดภัยมากขึ้น อย่างไรก็ตามงานวิจัยดังกล่าวมีข้อบกพร่องและข้อจำกัดดังนี้ กล่าวคือ การส่งข้อมูลที่สำคัญบางอย่างที่มีลักษณะเป็น Cleartext ทำให้สามารถถูกดักจับระหว่างทางได้ ถึงแม้ว่าจะมีการใช้ฟังก์ชันแฮช (Hash function) แล้วก็ตาม นอกจากนี้ยังขาดคุณสมบัติที่ UA สามารถยืนยันตัวของ Server ว่า เป็น Server ที่ UA ต้องการติดต่อด้วยจริงๆ

งานวิจัยฉบับนี้เสนอโพรโตคอลสำหรับการส่งข้อมูลระหว่าง UA และ Server ของโพรโตคอล SIP อย่างมั่นคงปลอดภัย กระบวนการภายในโพรโตคอลที่นำเสนอรวมไปถึงการกระจายและการตกลงคีย์ (Key Agreement) การเข้ารหัสลับข้อมูล (Encryption) และการยืนยันตัวตน (Authentication) พร้อมด้วยการแก้ไขปัญหาและข้อจำกัดของงานวิจัยที่มีอยู่ โดยนำเทคนิคการสร้างเซสชั่นคีย์แบบออฟไลน์[1] มาประยุกต์ร่วมด้วยโดยจะมีการใช้การเข้ารหัสลับโดยการใช้เซสชั่นคีย์ (Session Key) เพื่อเพิ่มความ

มั่นคงปลอดภัยในการเข้ารหัสลับ เซสชั่นคีย์ที่ใช้ในการติดต่อสื่อสารจะถูกแฮชพร้อมกับค่าของยูสเซอร์เนม (Username), พาสเวิร์ด (Password) ค่านอนซ์ (Nonce) ที่ใช้ในการป้องกันการโจมตีแบบ Replay Attack, และค่า N_A ของ UA ที่มีการสร้างทุกครั้งที่กระบวนการแลกเปลี่ยนคีย์ระหว่างกัน และ เซสชั่นคีย์ที่ใช้ในการติดต่อสื่อสารจะมีการเปลี่ยนแปลงทุกครั้งที่มีการใช้งานเซสชั่นคีย์หรือเมื่อเซสชั่นคีย์นั้นถูกโจมตีด้วยกระบวนการต่างๆ โดยการกระจายเซสชั่นคีย์จะแบบออฟไลน์ นั่นคือ จะไม่มีการส่งเซสชั่นคีย์ระหว่าง UA และ Server ซึ่งจะเป็นการสร้างความมั่นคงปลอดภัยในการติดต่อสื่อสาร

โครงสร้างของงานวิจัยฉบับนี้มีดังต่อไปนี้ บทที่ 2 กล่าวถึงพื้นฐานที่เกี่ยวข้อง บทที่ 3 นำเสนอโพรโตคอลสำหรับการส่งข้อมูลระหว่าง UA และ Server ของโพรโตคอล SIP อย่างปลอดภัย บทที่ 4 การวิเคราะห์คุณสมบัติความมั่นคงปลอดภัย บทที่ 5 สรุปผลการวิจัย

2. ทฤษฎีที่เกี่ยวข้อง

2.1 Session Initiation Protocol

Session Initiation Protocol หรือ SIP [6] เป็นโพรโตคอลที่มีการกำหนดมาตรฐานจาก IETF ซึ่งในปัจจุบันมีการนำไปใช้ในการควบคุมสัญญาณในส่งข้อมูลการสื่อสารระหว่างระบบเครือข่าย เช่น เสียง, ภาพเคลื่อนไหว บนเครือข่าย IP SIP มีหน้าที่หลัก คือ การสร้างการเชื่อมต่อ, การเปลี่ยนแปลงการเชื่อมต่อ และการยุติการเชื่อมต่อระหว่าง 2 ฝั่งหรือมากกว่านั้น ที่มีหนึ่งการเชื่อมต่อ หรืออาจจะมากกว่าหนึ่งการเชื่อมต่อก็ได้ การเปลี่ยนแปลงนี้อาจจะเกี่ยวกับ การเปลี่ยน ที่อยู่หมายเลข IP, หมายเลข ports ที่ใช้, การเพิ่มหรือลดคู่สายสนทนา เป็นต้น

โพรโตคอล SIP นั้นถูกออกแบบ [7]โดย Henning Schulzrinne และMark Handley ในปี 1996 รายละเอียดทางเทคนิคล่าสุดของ SIP คือ RFC3261 จาก IETF ในปี 2000 นั้นโพรโตคอล SIP ได้รับการยอมรับและเป็นส่วนหนึ่งใน 3GPP Signaling Protocol และถือได้ว่าอยู่ในสถาปัตยกรรม IP Multimedia Subsystem อย่างถาวร ซึ่งเป็นบริการของ IP-based Streaming multimedia ในระบบเซลลูล่าร์ (Cellular)

SIP เป็นโพรโตคอลที่อยู่ในระดับชั้นของแอพพลิเคชั่น (Application Layer) ซึ่งได้ถูกออกแบบให้เป็นอิสระจากระดับชั้นทรานสปอร์ต (Transport Layer) ที่อยู่ต่ำกว่า โพรโตคอล SIP มีลักษณะเป็น text-based protocol ซึ่งมีการใช้ส่วนประกอบร่วมกับโพรโตคอล HTTP และโพรโตคอล SMTP (Simple Mail Transfer Protocol)

การส่งข้อมูลของโพรโตคอล SIP นั้นมีรากฐานมาจากการรับส่งข้อมูลของโพรโตคอล HTTP ในการสื่อสารแต่ละครั้งไคลเอนท์ (Client) จะมีการส่งข้อความ HTTP Request และจึงรับการตอบกลับด้วยข้อมูล HTTP Response จาก HTTP Server จากพื้นฐานการส่งข้อมูลของโพรโตคอล HTTP นั้น โพรโตคอล SIP ได้นำเฮดเดอร์ (Header) การแปลงค่าและ Response code เกือบทั้งหมดของโพรโตคอล HTTP มาปรับใช้ในรูปแบบของโพรโตคอล SIP

โพรโตคอล SIP [2] ประกอบด้วยองค์ประกอบหลัก 2 ส่วน คือ

1. User Agent (หรือ UA) ทำหน้าที่ในการสร้างและรับข้อความในรูปแบบของ SIP นอกจากนั้นแล้วยังทำการจัดการเซสชั่นของ SIP อีกด้วย ซึ่ง UA เองอาจจะเป็นได้ทั้ง User Agent Client หรือ User Agent Server ซึ่งขึ้นอยู่กับการรับและส่งข้อความ

2. Proxy Server (หรือ Server) ทำหน้าที่ในการเป็นตัวกลางในการส่งข้อความระหว่าง UA หรือ Server อื่น ซึ่งจะทำการดูในส่วนของการเร้าติ้ง (Routing) และการตรวจสอบ UA นั้นๆ

2.2 ปัญหาทางด้านความมั่นคงปลอดภัยของโพรโตคอล SIP

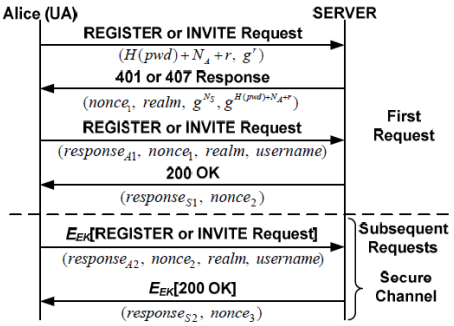
ในปัจจุบันนี้โพรโตคอล SIP ได้ถูกใช้อย่างแพร่หลาย เช่น VoIP และมัลติมีเดีย เป็นต้น ซึ่งถือได้ว่าเป็นหัวใจหลักต่อไปในอนาคต การปรับเปลี่ยนระบบโทรศัพท์พื้นฐาน (Plain Old Telephone Server หรือ POTS) มาเป็น IP-Based ซึ่ง SIP จะต้องเป็นตัวขับเคลื่อน [5, 6] ปัญหาด้านความมั่นคงปลอดภัยของโพรโตคอล SIP คือ กลไกการป้องกันที่ยังมีความหละหลวม เช่น

- HTTP Digest [3,6] นั้นยังไม่เพียงพอ ให้การตรวจสอบเพียงด้านเดียวเท่านั้นและยังมีความเสี่ยงที่จะถูกโจมตีแบบ Dictionary Attack และMan-In-The-Middle Attack
- โพรโตคอล TLS นั้นยังไม่อาจที่จะให้ความมั่นคงปลอดภัยที่ชัดเจน ถ้าต้องมีการผ่าน node มากกว่าหนึ่ง node และอาจจะมีความเสี่ยงที่เกิดการโจมตีแบบ DoS ได้

2.3 งานวิจัยที่เกี่ยวข้อง

2.3.1 โพรโตคอลของ Choi et al.

Choi *et al.* ได้มีการนำเสนอโพรโตคอลที่มีลดภาระการคำนวณคีย์ทางฝั่ง User Agent และมีการใช้วิธีการแลกเปลี่ยนคีย์แบบ DH half-key มาทำการเพิ่ม string เพื่อใช้เป็นค่า AK, IK, และEK ในการสื่อสารข้อมูลระหว่าง UA และ Server เทคนิคนี้จะเป็นการนำเสนอโพรโตคอลที่มีความมั่นคงปลอดภัยและรับรองความถูกต้องของ UA และ Server ที่ใช้ในการติดต่อสื่อสาร โดยมีการใช้ฟังก์ชั่นแฮชเพื่อเป็นการลดขนาดของข้อมูลที่ใช้ในการติดต่อสื่อสาร ซึ่งมีหลักการดังรูปที่ 1



รูปที่ 1. โพรโตคอลของ Choi *et al.*

1. Alice (UA) ทำการเลือกค่าสุ่มขึ้นมาหนึ่งค่า คือ r แล้วจึงนำไปผ่านกนะบวนการคำนวณค่า $g^{H(pwd)} \bmod p$ และ $g^r \bmod p$ หลังจากนั้นแล้วทำการเก็บค่าพารามิเตอร์ 3 ค่านั้นจากสมการข้างต้น

2. Alice (UA) ได้ทำการสร้างค่าสุ่มขึ้นมา โดยการคำนวณค่า *H(pwd)* + N_A + *r* ค่า N_A นั้นได้มีการสร้างขึ้นใหม่เมื่อ Alice (UA) ต้องการเริ่มทำการตรวจสอบและตกลงใช้คีย์ระหว่างกันในรอบใหม่ค่า $g^{H(pwd)} \bmod p$ และ $g^r \bmod p$ นั้นจะถูกนำกลับมาใช้ใหม่เพื่อลดภาระในการคำนวณของ DH half-key ที่ Alice (UA)

3. ในขณะที่มีการส่งข้อความ (REGISTER (or INVITE)) นั้น Server ทำการคำนวณค่า $g^{H(pwd)+N_A+r} \bmod p$ และสร้างค่าจากการคำนวณ DH half-key $g^{N_A} \bmod p$ ของ Alice (UA) โดยใช้กระบวนการตรวจสอบรหัสผ่านของผู้ใช้ $g^{H(pwd)} \bmod p$ ที่ถูกเก็บอยู่บน Server ตาม (1) หลังจากนั้น Server จะทำการสร้างค่าสุ่มขึ้นมา คือ N_S และทำการคำนวณ DH half-key $g^{N_S} \bmod p$ หลังจากนั้น Server จะทำการส่งข้อความ 401 (หรือ 407) ไปยัง Alice (UA)

$$g^{N_A} \bmod p = \frac{g^{H(user,pwd,N_A)+r} \bmod p}{g^{H(user,pwd,N_A)} \bmod p + g^r \bmod p} \qquad (1)$$

4. เมื่อ Alice (UA) ได้รับข้อความนั้น ก็นำไปผ่านกระบวนการสร้างคีย์หลัก (MK) $(g^{N_S})^{N_A} \bmod p$ สำหรับการตรวจสอบ และความมั่นคงปลอดภัยระหว่างกัน กล่าวคือ คีย์หลัก (MK) นั้นได้ถูกนำไปเป็นค่าเริ่มต้นในการสร้างคีย์ที่ใช้ในการตรวจสอบ (AK), คีย์ที่ใช้ในการป้องกันการเปลี่ยนแปลงของข้อมูล (IK), และคีย์ที่ใช้ในการเข้ารหัสข้อมูล (EK) โดยคีย์ที่กล่าวมานั้นได้มีการนำไปผ่านกระบวนการแฮชกับชุดของข้อความเฉพาะ ในขั้นตอนนี้ UA มีการคำนวณแค่ครั้งเดียวสำหรับการสร้างค่าคีย์หลัก ในเมื่อได้มีการแบ่งเบาภาระการคำนวณแบบ DH half-key บางส่วน $g^{N_A} \bmod p$ ไปยัง Server โดยที่ได้มีนำคีย์หลัก ไปสร้างเป็นพารามิเตอร์คีย์ที่ใช้ในการตรวจสอบ, คีย์ที่ใช้ในการป้องกันการเปลี่ยนแปลงของข้อมูล, และคีย์ที่ใช้ในการเข้ารหัสข้อมูล ตาม (2) คีย์ที่ใช้ในการตรวจสอบและรหัสผ่านของผู้ใช้ ได้ถูกนำไปใช้ในการตรวจสอบผู้ใช้ ระหว่างกระบวนการสร้าง HTTP Digest, คีย์ที่ใช้ในการป้องกันการเปลี่ยนแปลงของข้อมูล และคีย์ที่ใช้ในการเข้ารหัสข้อมูล นั้นได้ถูกใช้ในการปกป้องการรับส่งข้อความระหว่าง UA และServer

AK = H(MK, “AuthenticationKey”)

IK = H(MK, “IntegrityKey”) EK = H(MK, “EncryptionKey”) (2)

UA ได้ทำการคำนวณที่มีการรับรอง response_{A1} โดยใช้คีย์ที่ใช้ในการตรวจสอบ และรหัสผ่านของผู้ใช้ตาม (3) และส่งข้อความในรูปแบบของ SIP โพรโตคอลไปยัง Server หลังจาก Server ได้รับข้อความ Server ได้ทำการเก็บค่า DH half-key ของ UA และทำการคำนวณคีย์หลัก $(g^{N_S})^{N_A} \bmod p$ หลังจากนั้นจึงเกิดพารามิเตอร์ทั้ง 3 ตัวตาม (2) การใช้คีย์ที่ใช้ในการตรวจสอบนั้น Server จะสามารถตรวจสอบ UA กระบวนการตรวจสอบรหัสผ่านของผู้ใช้ สุดท้ายเมื่อทุกกระบวนการเสร็จสมบูรณ์ Server ได้ทำการสร้างค่าที่มีการรับรอง response_{S1} สำหรับการตรวจสอบตาม (3) เพื่อที่ Server สามารถตรวจสอบ UA ในครั้งต่อไป Server ได้สร้าง nonce_2 แล้วจึงส่งข้อความ (200 OK) พร้อมด้วยข้อมูลที่จำเป็นแก่ UA สุดท้าย UA ได้ทำการตรวจสอบข้อความที่ได้รับมา

$$\text{response}_{A1} = H(AK \,\|\, g^{H(pwd)}, nonce_1 \,\|\, realm \,\|\, g^{N_S} \,\|\, g^{N_A})$$

$$\text{response}_{S1} = H(AK \,\|\, g^{H(pwd)}, nonce_1 \,\|\, nonce_2 \,\|\, realm \,\|\, g^{N_S} \,\|\, g^{N_A}) \qquad (3)$$

เมื่อ UA ต้องการติดต่อกับ Server, UA และServer จะทำการสร้างค่าที่มีการรับรอง response_{A2} และ response_{S2} โดยใช้ คีย์ที่ใช้ในการตรวจสอบ, password และค่านอนซ์ตัวถัดไป ($nonce_2$) ตาม (4) ยิ่งไปกว่านั้นข้อความ SIP จะถูกปกป้องด้วยคีย์ที่ใช้ในการป้องกันการเปลี่ยนแปลงของข้อมูล และคีย์ที่ใช้ในการเข้ารหัสข้อมูล ระหว่างการส่งข้อมูลในช่วงเวลานั้น หลังจากที่การเชื่อมต่อความมั่นคงปลอดภัยหมดอายุนั้น UA เริ่มทำการสร้าง $N_A^{'}$ และทำการส่งค่าใหม่ H(pwd) + $N_A^{'}$ + r กับ $g^r \bmod p$ ไปเก็บที่ UA ซึ่งหมายวามว่า UA ยังคงทำการคำนวณแค่เพียงครั้งเดียวในการสร้างคีย์หลัก

$$\text{response}_{A2} = H(AK \,\|\, g^{H(pwd)}, nonce_2 \,\|\, realm \,\|\, username)$$

$$\text{response}_{S2} = H(AK \,\|\, g^{H(pwd)}, nonce_2 \,\|\, nonce_3 \,\|\, realm \,\|\, username) \quad (4)$$

2.3.2 เทคนิคการสร้างเซสชั่นคีย์แบบออฟไลน์ ของ Kungpisdan et al.

Kungpisdan *et al*. [1] ได้นำเสนอวิธีการสร้างคีย์แบบออฟไลน์ ซึ่งมีการสร้างเซสชั่นคีย์ (Session Key) โดยที่ไม่ต้องส่งคีย์ดังกล่าวผ่านเครือข่าย รายละเอียดของวิธีการดังกล่าวมีดังนี้

<u>การสร้างเซสชั่นคีย์ (Session Key Generation)</u>

UA และServer ได้ทำการใช้ค่าคีย์หลัก (MK) จากวิธีการตกลงการใช้คีย์ร่วมกันแบบ DH half-key กัน โดยจะนำคีย์หลักมาทำการแบ่งออกเป็น K_{AB} เรียกว่า long-term key, DK คือ Distribution key และ m คือ ค่าสุ่มที่ระบุจำนวนของคีย์ที่ต้องการสร้างขึ้น UAและ Serverได้สร้างเซตของ Preference keys K_i, i=1, 2,…, m ดังสมการ

$$K_i = h(K_{i-1}, DK) \quad (1)$$

หลังจากนั้น ทั้งคู่สร้างเซตของ Intermediate Keys ซึ่งเป็นการเพิ่มความยากในการวิเคราะห์การถอดรหัส เพิ่มความยากในการสืบค้นของ Preference key โดยมีรูปแบบดังนี้

$$IK^x_j = h(\text{conc}(IK^{x-1}_{mid}), IK^x_{j-1}) \quad (2)$$

โดย conc(IK^{x-1}_{mid}) จะเป็นการเชื่อมต่อ IK^{x-1}_{mid1}, IK^{x-1}_{mid2}, IK^{x-1}_{mid3}, … ตามลำดับ การใช้ Intermediate Keys ในทุกๆ รอบ จะทำการลบค่าออกจากระบบ ส่วนที่เหลือของ Intermediate Keys ในรอบอื่นๆ สามารถเขียนได้ดังนี้

$$\{K_1, K_2, \ldots, K_{rm}\},$$
$$\{K^1_1, K^1_2, \ldots, K^1_{rm}\},$$
$$\{K^2_1, K^2_2, \ldots, K^2_{rm}\},$$
$$\ldots$$
$$\{K^n_1, K^n_2, \ldots, K^n_{rm}\}$$

ผลที่ได้ในรอบสุดท้ายของ Intermediate Keys ที่ได้พิจารณาจาก เซสชั่นคีย์ (Session key) (SK_j) โดยที่ j = 1, … , m คือ

$$IK^n_1 = SK_1, IK^n_2 = SK_2, \ldots, IK^n_m = SK_m \quad (3)$$

จากเทคนิคนี้จะเห็นว่าจะมีการส่งตัวแปลที่เป็นค่าเริ่มต้นเพื่อนำไปใช้ในการสร้างคีย์ และคีย์ที่ถูกสร้างขึ้น ถูกนำมาใช้เพียงครั้งเดียวไม่มีการใช้ซ้ำ ในบทความวิจัยนี้จะมีการนำเอาเทคนิคการกระจายคีย์ดังกล่าวมาใช้ เพื่อเพิ่มความมั่นคงปลอดภัยให้กับโพรโตคอล SIP ซึ่งรายละเอียดของเทคนิคที่นำเสนอนั้นจะกล่าวในบทถัดไป

3. งานวิจัยที่นำเสนอ

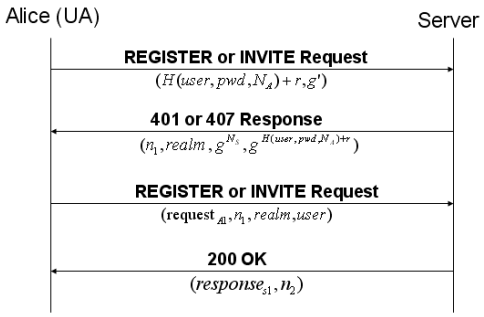
เพื่อเป็นการแก้ไขปัญหาและข้อจำกัดงานงานวิจัยที่มีอยู่ [3] งานวิจัยฉบับนี้โพรโตคอลสำหรับการส่งข้อมูลระหว่าง User Agent และServer ของ SIP โพรโตคอลอย่างปลอดภัย โดยมีการประยุกต์นำเอาเทคนิคการสร้างเซสชั่นคีย์แบบออฟไลน์ [1], การตกลงการใช้คีย์แบบ DH half-key [3], การตรวจสอบทั้ง 2 ฝั่ง [1] ดังรายละเอียดด้านล่าง

3.1 นิยามศัพท์ที่เกี่ยวข้อง

- {user, pwd} คือ เซ็ทของยูสเซอร์เนม และพาสเวิร์ดของ User Agent
- MK คือ ค่าเริ่มต้นในการสร้างเซสชั่นคีย์ (คีย์หลัก)
- H(m) แทนข้อความที่ผ่านการแฮชฟังก์ชั่น (Hash function)
- n แทนค่าอนซ์ (nonce) ใช้ป้องกันการรีเพลย์ (Replay Attack)
- r แทนค่าสุ่มของ User Agent
- N_A แทนค่าที่ User Agent เพื่อใช้ในการทำการแลกเปลี่ยนคีย์ในแต่ละรอบ
- N_S แทนค่าที่ Server เพื่อใช้ในการทำการแลกเปลี่ยนคีย์ในแต่ละรอบ
- SK^{UA}_j แทนค่าเซสชั่นคีย์ (Session Key) ที่ถูกสร้างขึ้น

3.2 รายละเอียดของโพรโตคอลที่นำเสนอ

ระบบนี้ประกอบด้วยผู้ที่เกี่ยวข้องจำนวน 2 ส่วน คือ User Agent และ Server ในตอนแรกทั้ง 2 ฝั่งได้ทำการแลกเปลี่ยนคีย์กันโดยใช้กรรมวิธีแบบ DH half-key หลังจากนั้นมีการนำผลลัพธ์มาสร้างเป็นคีย์หลัก (MK) เพื่อนำไปใช้ในการสร้างเซสชั่นคีย์ต่อไป โดยการสร้างเซสชั่นคีย์นั้นได้ใช้เทคนิคการสร้างเซสชั่นคีย์แบบออฟไลน์ของ Kungpisdan *et al*. [1] ซึ่งมีความมั่นคงปลอดภัยในการดำเนินงานเป็นอย่างมาก เนื่องจากเซสชั่นคีย์ที่ถูกสร้างขึ้นจะถูกใช้เพียงครั้งเดียว ไม่มีการใช้ซ้ำเมื่อได้เซสชั่นคีย์แล้ว หลังจากนั้นมีการนำค่าเซสชั่นคีย์นั้นมาผ่านฟังก์ชั่นแฮช เพื่อนำไปใช้ในกระบวนการส่งข้อมูลและตรวจสอบทั้งฝั่ง User Agent และServer



รูปที่ 2. ขั้นตอนตกลงการสร้างคีย์ระหว่าง User Agent และ Server

1. Alice (UA) จะทำการเลือกค่าสุ่มขึ้นมาสองค่า คือ r และ N_A แล้วจึงมาทำการคำนวณค่า $g^{H(user,pwd,N_A)} \bmod p$ และ $g^r \bmod p$ หลังจากนั้นได้ทำการเก็บค่าพารามิเตอร์ 4 ตัวที่ได้ตามข้างต้น หลังจากนั้นมีการคำนวณค่า H(user, pwd, N_A) + r ค่า N_A นั้นได้มีการสร้างใหม่เมื่อ Alice (UA) ต้องการตกลงใช้คีย์ระหว่างกันในรอบใหม่ ค่า $g^{H(user,pwd,N_A)} \bmod p$ และ $g^r \bmod p$ นั้นได้ถูกนำกลับมาใช้ใหม่เพื่อลดภาระในการคำนวณของ DH half-key ที่ Alice (UA)

2. ในขณะที่มีการส่งข้อความ (REGISTER (or INVITE)), Server ได้มีการคำนวณค่า $g^{H(user,pwd,N_A)+r} \bmod p$ และมีการสร้างค่าของ Alice (UA) DH half-key $g^{N_A} \bmod p$ โดยกระบวนการตรวจสอบรหัสผ่านของผู้ใช้ $g^{H(user,pwd,N_A)} \bmod p$ ที่ถูกเก็บอยู่บน Server (1) หลังจากนั้น Server ได้ทำการสร้างค่าสุ่มขึ้นมา คือ N_S และทำการคำนวณของ DH half-key $g^{N_S} \bmod p$ หลังจากนั้น Server ได้ทำการส่งข้อความ 401 (หรือ 407) เพื่อแจ้ง Alice (UA) เมื่อการคำนวณเสร็จสิ้น

$$g^{N_A} \bmod p = \frac{g^{H(user,pwd,N_A)+r} \bmod p}{g^{H(user,pwd,N_A)} \bmod p + g^r \bmod p} \qquad (1)$$

3. เมื่อ Alice (UA) ได้รับข้อความตอบรับจาก Server ก็มีการนำค่าที่ได้ไปสร้างเป็นคีย์หลัก $(g^{N_S})^{N_A} \bmod p$ เป็นค่าเริ่มต้นในการสร้างเซสชั่นคีย์ ในขั้นตอนนี้ Alice (UA) ได้ทำการคำนวณเพื่อสร้างคีย์หลักแค่ครั้งเดียวเท่านั้น เนื่องจาก Server นั้นได้ทำการคำนวณในส่วนของ $g^{N_A} \bmod p$ สำหรับ DH half-key หลังจากนั้นได้นำคีย์หลักไปสร้างเซสชั่นคีย์ โดยใช้เทคนิคการสร้างเซสชั่นคีย์แบบออฟไลน์ของ Kungpisdan *et al.* [1]

4. Alice (UA) ได้ทำการคำนวณค่าที่มีการรับรอง request_{A1} โดยมีการใช้ค่าเซสชั่นคีย์ SK_j^{UA}, ชื่อผู้ใช้, รหัสผ่าน ตาม (2) และส่งข้อความไปยัง Server หลังจากนั้นเมื่อ Server ได้รับข้อความ ก็ได้ทำการตรวจสอบค่า request_{A1} หลังจากการตรวจสอบแล้วเสร็จ ถ้าค่าแฮชนั้นตรงกันก็ทำการส่งข้อความตอบรับ (200 OK) กลับไปยัง Alice (UA) พร้อมด้วยการส่งค่าที่ผ่านการคำนวณพร้อมด้วยการรับรอง response_{S1} และค่านอนซ์เพื่อให้เป็นการตรวจสอบทั้ง User Agent และServer

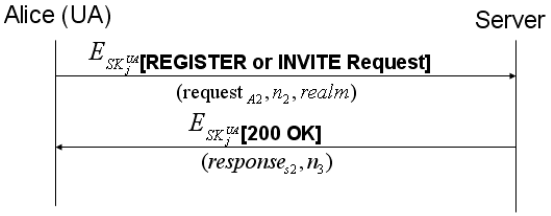
$$\text{request}_{A1} = H(SK_j^{UA} \| g^{H(user,pwd,N_A)}, n_1 \| realm \| g^{N_s} \| g^{N_A})$$

$$\text{response}_{S1} = H(SK_j^{UA} \| g^{H(user,pwd,N_A)}, n_1 \| n_2 \| realm \| g^{N_s} \| g^{N_A}) \qquad (2)$$

5. ในครั้งต่อไปเมื่อ Alice (UA) ต้องการติดต่อกับ Server ก็ได้ทำการส่งข้อความ (REGISTER (or INVITE)) หลังจากนั้นได้ทำการสร้างค่าที่มีการรับรอง request_{A2} และ response_{S2} ตาม (3) โดยมีการใส่เซสชั่นคีย์แล้วได้มีการนำไปผ่านกระบวนการแฮช โดยในระหว่างการส่งข้อมูลตลอดอายุของเซสชั่นคีย์นั้น การส่งข้อมูลได้รับการปกป้องอย่างปลอดภัย โดยที่การส่งข้อมูล 1 ครั้งนั้นจะมีกระบวนการคำนวณโดยการบวกเพิ่มขึ้น 1 ครั้ง

$$\text{request}_{A2} = H(SK_j^{UA} \| g^{H(user,pwd,N_A)}, n_2 \| realm)$$

$$\text{response}_{S2} = H(SK_j^{UA} + 1 \| g^{H(user,pwd,N_A)}, n_2 \| n_3 \| realm) \qquad (3)$$



รูปที่ 3. ขั้นตอนการสื่อสารระหว่าง User Agent และ Server โดยใช้ Session Key

4. การวิเคราะห์คุณสมบัติความมั่นคงปลอดภัย

4.1 Confidentiality

จากการที่ มีการสื่อสารกันด้วยเซสชั่นคีย์ (SK_j^{UA}) นั้น ผู้ที่มีคีย์ SK_j^{UA} ที่ตรงกันเท่านั้น จึงสามารถเข้าถึงหรือติดต่อสื่อสารกัน ระหว่าง UA และ Server ได้ ซึ่งการสร้างเซสชั่นคีย์นั้นเป็นแบบออฟไลน์ ซึ่งจะไม่มีการส่งคีย์

ไปมานั้น จึงทำให้โอกาสในการถูกดักจับ (Sniff) แล้วได้ค่าคีย์ที่ถูกต้องไป ทำได้ยากมาก จึงทำให้การเข้ารหัสคีย์มีความมั่นคงปลอดภัยมากยิ่งขึ้น

4.2 Message Authentication

เป็นการใช้สำหรับการยืนยันความถูกต้องของข้อมูลระหว่างการสื่อสารและป้องกัน การปลอมข้อมูล หากพิจารณาจากโพรโตคอลข้างต้นแล้ว $H(SK_j^{UA} \| g^{H(user,pwd,N_A)}, n_1 \| realm \| g^{N_S} \| g^{N_A})$ จะพบว่า ข้อความที่มีการส่งผ่านระหว่างกันนั้น จะใช้เซสชั่นคีย์และ ค่านอนซ์เป็นหลักเพื่อพิสูจน์ว่าข้อมูลที่ใช้ในการสื่อสาร (Message Authentication Code: MAC) มาจาก UA และ Server ที่สามารถเชื่อถือได้

4.3 การทนทานต่อการถูกโจมตีด้วย Replay Attack

การโจมตีแบบ Replay Attack [8] คือ การที่แพ็คเกจของข้อมูลที่มีความถูกต้องได้ถูกทำให้ล่าช้า เนื่องมาจากข้อมูลนั้นอาจถูกทำการแก้ไข หรือปลอมแปลง แล้วจึงมีการส่งออกมาใหม่ แม้ว่าจะถูกการโจมตีแบบ Replay Attack ด้วยวิธีการต่างๆ เช่น การปลอมตัวเป็น UA หรือ การปลอมตัวเป็น Server การโจมตีดังกล่าวนั้นไม่สามารถประสบความสำเร็จได้เนื่องจากเซสชั่นคีย์มีการเปลี่ยนไปตลอดเวลา เมื่อมีการติดต่อกันอย่างสมบูรณ์ ระหว่าง UA และ Server การใช้ค่านอนซ์ (n_1, n_2) ก็เป็นส่วนสำคัญในการป้องกัน เนื่องจากจะเป็นการแสดงค่า เพื่อบ่งบอกว่าเกิดขึ้นจาก UA หรือServer โดยแท้จริง

4.4 การทนทานต่อการถูกโจมตีแบบ Brute Force Attack

ด้วยเทคนิคการสร้างคีย์แบบออฟไลน์ (IK^x_j = h(conc(IK^{x-1}_{mid}), IK^x_{j-1})) [1] และเทคนิคการกระจายคีย์แบบ DH half-key ทำให้การสืบค้นเพื่อหา Preference key ทำได้ยากขึ้น เนื่องจาก เป็นการเข้ารหัสที่มีความซับซ้อน และค่าของ IK^x_j หรือ จะมีการเปลี่ยนไปเรื่อยๆ จึงทำให้โอกาสที่การโจมตีแบบ Brute Force ไม่ประสบผลสำเร็จ

5. สรุปผลการวิจัย

งานวิจัยฉบับนี้ได้นำเสนอรูปแบบการติดต่อสื่อสารระหว่าง User Agent และ Server เพื่อเกิดความความมั่นคงปลอดภัยในด้านต่างๆ โดยการสื่อสารในรูปแบบการเข้ารหัสลับแบบสมมาตร (Symmetric Encryption) ซึ่งอาศัยวิธีการแลกเปลี่ยนคีย์แบบ DH half-key และมีการสร้างเซสชั่นคีย์แบบออฟไลน์ โดยที่ไม่ต้องส่งคีย์ดังกล่าวผ่านเครือข่าย การนำเสนอนี้ สามารถใช้ได้กับ User Agent ที่มีขนาดเล็กได้โดยที่ไม่เพิ่มภาระมากนัก และยังช่วยลดปริมาณของข้อมูลที่ส่งระหว่างเครือข่ายได้อีกด้วย

การพัฒนาต่อไปนั้น จะมีการไปนำไปประยุกต์และพัฒนาใช้ในส่วนของการใช้งานระหว่างโดเมน โดยที่ยังไม่เห็นระบบความปลอดภัยที่เป็นรูปธรรม มีแค่เพียงระบบความปลอดภัยเบื้องต้นเท่านั้น ซึ่งถ้ามีการนำแบบแผนนี้ไปปรับใช้ ก็สามารถทำให้เกิดการกำหนดมาตรฐานความปลอดภัยระหว่างโดเมนขึ้นได้

เอกสารอ้างอิง

[1] Kungpisadan *et al.* “A Secure Offline Key Generation With Protection Against Key Compromise”, Proceedings of the 13th World Multi-conference on Systemics, Cybernetric, and Informatics, pp. 63-67, 2009.

[2] IETF Standard, SIP: Session Initiation Protocol, IETF RFC 3261, Jun. 2002.

[3] J. Choi, S. Jung, K. Bae, and H. Moon, “A Lightweight Authentication and Hop-by-by-Hop Security Mechanism for SIP network”, Advanced Technologies for Communications, 2008. ATC 2008. International Conference on, 236, 6-9 Oct. 2008.

[4] Kungpisdan, S.; Thai-udom, T. “Securing micropayment transactions over Session Initiation Protocol”, Communications and Information Technology, 2009. ISCIT 2009. 9th International Symposium on, 190, 28-30 Sept. 2009.

[5] E. Cha, H. Choi, S. Cho, “Evaluation of Security Protocols for the Session Initiation Protocol”, Computer Communications and Networks, 2007. ICCCN 2007. Proceedings of 16th International Conference on, 611-616, 13-16 Aug. 2007.

[6] C. Tao, G. Qiang, H. Baohong, “A Lightweight Authentication Scheme for Session Initiation Protocol”, Communications, Circuits and Systems, 2008. ICCCAS 2008. International Conference on, 502-505, 25-27 May 2008.

[7] Wikipedia.“Session Initiation Protocol”. en.wikipedia.org/wiki/Session_Initiation_Protocol.2010

[8] Wikipedia. “Replay Attack”. en.wikipedia.org/wiki/Replay_attack. 2010