การปรับปรุงรหัสผ่านใช้ครั้งเดียวแบบเอสคีย์โดยใช้การแปลงรหัสข้อมูลแบบฐาน 64

An Improvement of the S/key One Time Password (OTP) using a BASE64 Data Encodement

ปรัชญา ไชยเมือง (Pratchya Chaimuange)*, สุชาติ คุ้มมะณี (Suchart Khummanee)*
สมนึก พ่วงพรพิทักษ์ (Somnuk Puangpronpitag)*

บทคัดย่อ

ในปัจจุบันรหัสผ่านใช้ครั้งเดียวเป็นวิธีการที่สำคัญมาก ทำให้กระบวนการยืนยันตัวตนปลอดภัยจากการโจมตีโดยการดักจับข้อมูลรหัสผ่านและนำกลับมาใช้ใหม่ ซึ่ง S/Key ได้ถูกเสนอให้เป็นหนึ่งในมาตรฐานของรหัสผ่านใช้ครั้งเดียว อย่างไรก็ตาม ความยาวของ OTP ที่แสดงผล แบบ S/Key ยังไม่เหมาะสม ดังนั้นในงานวิจัยนี้จึงได้ออกแบบ BASE64 S/Key OTP ที่มีฐานการสร้าง OTP แบบ S/Key และใช้การแปลงรหัสแบบ BASE64 นอกจากนี้ BASE64 S/KEY OTP ยังได้เปรียบเทียบการทำงาน กับ S/Key OTP, HOTP และ TOTP ผลการประเมินชี้ให้เห็นถึงความมีประสิทธิผลที่เพิ่มขึ้นจากการใช้ BASE64 S/KEY OTP

คำสำคัญ: รหัสผ่านใช้ครั้งเดียว, รหัสผ่าน, การโจมตีด้วยการสุ่มรหัสผ่านทีละตัวจนครบทุกข้อมูลที่เป็นไปได้

Abstract

Nowadays, One Time Password (OTP) is very significant to provide an authentication process that is secure against attacks based on replaying captured reusable password. S/key OTP has been proposed as one of the OTP standards. However, the length of S/key OTP is still not appropriate. So, in this paper, we have proposed a new design of OTP based on the S/key OTP and a BASE64 data encodement. Furthermore, the new BASE64 S/key OTP has been compared with the S/Key OTP, HOTP, and TOTP. The evaluation results have demonstrated a more effectiveness of our BASE64 S/key OTP.

Keyword: One Time Password, Password, Brute-force Attack Password.

1. บทนำ

ปัจจุบันการยืนยันตัวตนผู้ใช้ในระบบสารสนเทศส่วนใหญ่ใช้รหัสผ่าน (Password) ซึ่งเป็นปัจจัยการยืนยันตัวตนแบบ User Knowledge [1] แต่การใช้รหัสผ่านเพียงอย่างเดียวมีจุดอ่อน คือ (1) รหัสผ่านรั่วไหลได้หลายวิธี (ถูกดักจับข้อมูล บอกต่อผู้อื่นโดยไม่ตั้งใจ เขียนไว้กันลืม รหัสผ่านเดาง่าย) (2) Login ด้วยรหัสผ่านตัวเดิมทุกครั้ง ทำให้เมื่อรหัสผ่านรั่วไหลไปจะถูกนำไปใช้ได้ทันทีโดยที่ระบบและผู้ใช้ตรวจสอบไม่ได้

One Time Password (OTP) [2] เป็นวิธีหนึ่งที่นำเสนอเพื่อแก้ไขปัญหาที่เกิดขึ้นโดยออกแบบให้รหัสผ่านเปลี่ยนแปลงทุกครั้งเมื่อ Login ซึ่ง OTP ในปัจจุบันมี 3 แบบ คือ HOTP [3], TOTP [4] และ S/Key [5] โดยแต่ละตัวแสดงผลของ OTP แตกต่างกัน จึงทำให้มีจุดอ่อนที่เกิดขึ้นดังนี้ HOTP และ TOTP แสดงผลเป็นตัวเลข 6-8 ตัวอักษร ซึ่งง่ายต่อการ Brute-force Attack [6] เพื่อเดาสุ่มรหัสผ่าน ส่วน S/Key แสดงผลมากถึง 24 ตัวอักษรทำ Brute-force Attack ได้ยากกว่าแต่มีปัญหาคือต้องพิมพ์ข้อมูลมากกว่า

และในปัจจุบัน (พ.ศ. 2554) ได้มีแนวคิด ในการใช้ User Possession [1] แบบต่างๆ ให้แสดงผล OTP เพื่อทำการ

* คณะวิทยาการสารสนเทศ มหาวิทยาลัยมหาสารคาม

ยืนยันตัวตนด้วยปัจจัยสองแบบ (Two Factor Authentication :TFA) [2] สำหรับแก้ปัญหาการรั่วไหลของรหัสผ่านที่เกิดขึ้น ตัวอย่างเช่น Gmail (ปี พ.ศ. 2554) ได้มีกระบวนการยืนยันตัวตนผู้ใช้ด้วย TFA จาก OTP และเช่นเดียวกับ Facebook.com (ปี พ.ศ. 2553) หรือแม้แต่ในระบบ E-Banking ของธนาคารก็ไม่ให้ความเชื่อถือต่อการใช้รหัสผ่านเพียงอย่างเดียว โดยในเดือน กรกฎาคม พ.ศ. 2553 มีนโยบายจากธนาคารแห่งประเทศให้ระบบ E-Banking ของธนาคารต้องมีปัจจัยการยืนยันตัวตน ผู้ใช้ในระบบมากกว่าหนึ่งปัจจัย แต่การใช้ OTP ในระบบดังกล่าวเลือกใช้ HOTP และ TOTP ซึ่งเสี่ยงต่อการทำ Brute-force Attack ได้ง่าย และถึงแม้จะมีการป้องกัน Brute-force Attack โดย Trial and Error Check [7] เพื่อกำหนดจำนวนครั้งในการพิมพ์รหัสผ่านผิด ซึ่งปัจจุบันคือ 3 ครั้ง หากการ Login มีการส่งรหัสผ่านผิดเกิน Trial and Error Check ที่กำหนด ระบบจะมองว่าถูก Brute-force Attack และยกเลิกการ Login แบบเดิมทันที จากนั้นระบบจะมีกระบวนการ Login แบบใหม่ที่เพิ่มการตรวจสอบว่า รหัสผ่านที่ส่งเพื่อ Login นั้น มาจากการพิมพ์ข้อมูลของมนุษย์จริงไม่ใช่มาจากบอทหรือโปรแกรมอัตโนมัติ ซึ่งเทคนิคที่นิยมใช้ป้องกันคือ CAPTCHA [8]

ด้วยความเสี่ยงจาก Brute-force Attack ของ HOTP และ TOTP ที่มีมาก นำไปสู่ Trial and Error Check ที่มีค่าน้อย นั้นคือจำนวนครั้งในการพิมพ์รหัสผ่านผิดของผู้ใช้น้อยตามไปด้วย ซึ่งทำให้การพิมพ์รหัสผ่านผิดจากผู้ใช้จริงถูกมองว่าเป็นการ Brute-force Attack ทำให้ผู้ใช้ต้องมีภาระเพิ่มขึ้นเพื่อตรวจสอบความเป็นมนุษย์จริง ซึ่งอาจจะส่งผลให้กระบวนการ Login ล่าช้าและไม่สำเร็จได้

งานวิจัยนี้จึงได้เสนอแนวคิดในการแก้ปัญหาใหม่ โดยเลือก OTP ที่มีความเสี่ยงต่อการ Brute-force Attack น้อยกว่า HOTP และ TOTP นั้นคือ S/Key โดยการปรับปรุงจำนวนตัวอักษร OTP แบบ S/Key ให้ลดลง ด้วยเทคนิคการแปลงรหัสข้อมูลแบบฐาน 64 (BASE64 Data Encoding) [9] และได้ออกแบบและพัฒนาโปรแกรมต้นแบบเพื่อแสดงผล OTP ดังกล่าว นอกจากนี้ ในงานวิจัยยังได้ประเมินผลการทดสอบโปรแกรมเพื่อแสดงให้เห็นถึงความสามารถในการแก้ปัญหาที่เกิดขึ้นได้จริง และสรุปผลการทดสอบ

ในส่วนที่ 2 ของเอกสารฉบับนี้ได้กล่าวถึงทฤษฎีและงานวิจัยที่เกี่ยวข้อง ส่วนที่ 3 กล่าวถึงแรงจูงใจของงานวิจัย ส่วนที่ 4 กล่าวถึงการออกแบบและพัฒนาระบบและส่วนที่ 5 และ 6 กล่าวถึง ผลการทดสอบและวิเคราะห์เปรียบเทียบ พร้อมสรุปผลการพัฒนาตามลำดับ

2. ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 รหัสผ่านใช้ครั้งเดียว (One Time Password: OTP)

OTP [2] คือ วิธีการหนึ่งที่แก้ปัญหาการใช้รหัสผ่าน ซึ่งมีแนวคิดให้รหัสผ่านที่ใช้ Login เปลี่ยนแปลงอยู่ตลอดทุกครั้งที่เข้าใช้งานระบบ มีมาตรฐานที่นิยมใช้งานกัน 3 แบบดังนี้

1) HOTP คือ OTP ที่จะเปลี่ยนแปลงไปตามตัวนับ (Counter) ซึ่งใช้หลักของ HMAC [10] คือ การ Hash ข้อมูลเริ่มต้น (Seed) ในที่นี้คือ Counter ด้วย Hash Function Algorithm (ใน HOTP ใช้ SHA-1) และนำค่า Hash ที่ได้แปลงเป็น OTP ด้วย HOTP Truncate Function (ตามเอกสาร RFC 4226 [3] หัวข้อ 5.3) ได้ผล OTP ในรูปแบบตัวเลข 6-8 หลัก

2) TOTP คือ OTP ที่จะเปลี่ยนแปลงไปตามเวลาที่เปลี่ยนไป โดยมีหลักการทำงานแบบเดียวกับ HOTP ต่างกันที่ Seed จากเดิมใช้ Counter เปลี่ยนเป็นใช้เวลา (Time) ปัจจุบัน และ Hash Function Algorithm ที่ใช้เปลี่ยนไปถึง 2 แบบ คือใช้ SHA-256, SHA-512 แต่การแสดงผลยังอยู่ในรูปแบบตัวเลข 6-8 หลัก เหมือนกับ HOTP

3) S/Key คือการสร้าง OTP ที่เปลี่ยนไปตามข้อมูล Challenge ที่ Server แสดง หาก Challenge เปลี่ยนไป OTP ที่ได้ก็จะเปลี่ยนตามไป โดยการทำงานของ S/Key จะคล้ายกับ HOTP โดยเปลี่ยน Seed เป็น Challenge ที่ Server แสดง และฟังก์ชั่นในการแสดงผล OTP ก็เปลี่ยนแปลงไป คือใช้ FoldTo64Bit Function (อธิบายไว้ใน Appendix A ของ RFC 2289 [5]) ซึ่งแสดงผลเป็นตัวอักษรจำนวน 6-24 ตัวอักษร (อธิบายการทำงานไว้ใน Appendix B ของ RFC 2289)

จาก OTP แต่ละแบบ สามารถนำวิเคราะห์ความเป็นไปได้ที่จะเกิดผลของ OTP ทั้งหมดได้ดังดังตารางที่ 1

ตารางที่ 1 *แสดงความเป็นไปได้ที่เกิดผลทั้งหมดของ OTP แต่ละแบบ*

OTP Type	Choice	Output	Probability (P)	Log10(P)	
HOTP/TOTP	10	6	10,000,000	7	10^7
HOTP/TOTP	10	7	100,000,000	8	10^8
HOTP/TOTP	10	8	1,000,000,000	9	10^9
S/Key	2048	6 [4-24]	73,786,976,294,838,200,000	19.86798	10^19

การวิเคราะห์ความเป็นได้หรือความน่าจะเป็น (Probability: P) ที่จะเกิดผลทั้งหมดของ OTP แต่ละแบบ เพื่อแสดงให้เห็นถึงความเสี่ยงที่จะถูก Brute-force Attack มีผลดังนี้ HOTP และ

TOTP แสดงข้อมูลเป็นตัวเลข (0-9) จำนวน 6-8 หลัก หาค่า P ได้จาก

$$P = (p_1 * p_2 * ..... p_n)$$

โดยที่ P คือ จำนวนความเป็นไปได้ทั้งหมด

P_1 คือความเป็นได้ที่จะเกิดอักษรหลักที่ 1

P_2 คือความเป็นได้ที่จะเกิดอักษรหลักที่ 2

P_n คือความเป็นได้ที่จะเกิดอักษรหลักที่ n

จะได้ $P_{(HOTP)} = (10*10*10*10*10*10)=10^6$ (ในกรณีที่ตัวเลข 6 หลัก ความเป็นไปได้ที่จะเกิดตัวอักษรแต่ละหลักคือ 10 นั้นคือ 0 ถึง 9) ดังนั้น HOTP และ TOTP ที่แสดงผลเป็นตัวเลข 6-8 หลัก จะมีความเป็นไปได้ที่จะเกิดผลของ OTP ทั้งหมดคือ 10^6 ถึง 10^8 ตัว และ เช่นเดียวกันกับการหา P ของ S/Key จะได้

$P_{(S/Key)} = (2048*2048*2048*2048*2048*2048) = 2048^6$ หรือประมาณ 10^{19} เนื่องการแสดงผลแบบ S/Key เป็นการสร้าง OTP จากคำภาษาอังกฤษตัวพิพม์ใหญ่ที่เตรียมไปไว้ทั้งหมด 2048 คำ โดยเลือกมาแสดง 6 คำ แต่ละคำมี 1- 4 ตัวอักษร (ความเป็นไปได้ที่จะเกิด OTP 1 คำ คือ 2048) จากข้อมูลข้างต้นจะเห็นได้ว่า ความเป็นไปได้ที่จะเกิดผลของ OTP แบบ S/Key มากกว่า HOTP และ TOTP เป็นอย่างมาก จึงทำให้ S/Key ปลอดภัยจาก Brute-force Attack มากกว่า แต่จุดอ่อนของ S/Key คือจำนวน Output ไม่คงที่และมีอาจได้มากถึง 24 ตัวอักษร ซึ่งจำนวนตัวอักษรที่มาก จะเพิ่มความผิดพลาดในการพิมพ์ OTP ของผู้ใช้ในขณะ Login ได้

2.2 Brute-force Attack Password

Brute-force Attack Password [6] คือ การโจมตีระบบเพื่อ Login แบบหนึ่ง ที่ใช้การสุ่มรหัสผ่านที่ละตัวไปจบครบทุกๆ รหัสผ่านที่เป็นไปได้ทั้งหมด เป็นวิธีการโจมตีที่ได้ผล 100 % หากมีเวลามากพอ ซึ่ง Brute-force Attack นั้นทำได้ง่ายหากความเป็นไปได้ของรหัสผ่านทั้งหมดมีน้อย เช่น ตั้งรหัสผ่านเป็นตัวเลขทั้ง 8 หลัก จะได้ความเป็นไปได้ทั้งหมดของการทำ Brute-force Attack คือ 10^8 ครั้ง และในทางกลับกัน Brute-force Attack จะใช้เวลามากขึ้น หากความเป็นไปได้ทั้งหมดที่จะเกิดรหัสผ่านมีมาก เช่น ตั้งรหัสผ่านที่ประกอบด้วยตัวอักษรภาษาอังกฤษ (ตัวพิพม์เล็ก+ตัวพิพม์ใหญ่) ทั้งหมด 8 หลัก ความเป็นไปได้ทั้งหมดของ Brute-force Attack คือ 48^8 ครั้ง และจากข้อมูลความเร็วในการ Brute-force Attack รหัสผ่านที่เป็นตัวเลข 8 หลัก ซึ่งดูจากความสามารถของเครื่องคอมพิวเตอร์กล่าวว่า ใน Pentium 100 ใช้เวลา 1 นาที 30 วินาทีและใน Dual Processor PC ใช้เวลา 10 วินาที [11] จากข้อมูลข้างต้นเห็นได้ว่าการ Brute-force Attack นั้นสามารถทำได้อย่างรวดเร็วหากเทียบกับความสามารถของเครื่องคอมพิวเตอร์ในปัจจุบัน จึงทำให้จำนวนความเป็นได้ที่จะเกิดรหัสผ่านทั้งหมดนั้นมีความสำคัญต่อความเสี่ยงที่จะเกิดจากการทำ Brute-force Attack

2.3 Trial and Error Check

การตรวจสอบการลองผิดลองถูก (Trial and Error Check) [7] เป็นกระบวนการหนึ่งที่สำคัญต่อการยืนยันตัวตนผู้ใช้ด้วยรหัสผ่าน โดยการกำหนดจำนวนครั้งในการลองผิดลองถูกที่ผู้ใช้จะพิมพ์รหัสผ่านในการ Login นั้นก็คือการกำหนดจำนวนครั้งที่ผู้ใช้จะพิมพ์รหัสผ่านผิดได้นั้นเอง ซึ่งวิธีการนี้นำมาใช้ป้องกันการ Brute-force Attack โดยหากมีการส่งรหัสผ่านที่ใช้ Login ผิดเกินค่า Trial and Error Check ที่กำหนดไว้ ระบบจะไม่อนุญาตให้บัญชีผู้ใช้นั้น Login ได้อีก แต่จะมีขั้นตอนในการตรวจสอบความเป็นมนุษย์จริงเพื่อตรวจสอบว่า Login นั้นเกิดขึ้นโดยผู้ใช้จริง ไม่ได้เกิดจากบอทหรือโปรแกรมประยุกต์ที่ใช้ทำ Brute-force Attack โดยขั้นตอนการตรวจสอบที่นิยมใช้งานมากที่สุดคือ แค๊ปท์ชา (CAPTCHA) [8] ซึ่งเป็นการทดสอบแบบโต้ตอบชนิดหนึ่ง เพื่อทดสอบว่าผู้ใช้งานเป็นมนุษย์จริงหรือไม่ ด้วยเหตุนี้การกำหนดค่า Trial and Error Check นั้นมีความสำคัญต่อการ Login คือหากกำหนดค่ามากความเสี่ยงที่จะถูก Brute-force Attack ก็มีมาก และหากค่าน้อยย่อมปลอดภัยมากกว่า แต่ปัญหาก็คือผู้ใช้จะสามารถพิมพ์รหัสผ่านผิดในจำนวนครั้งที่น้อยลง และหากพิมพ์ผิดเกิน Trial and Error Check จะต้องมีขั้นตอนในการตรวจสอบความเป็นมนุษย์จริง ซึ่งเป็นการเพิ่มภาระให้ผู้ใช้

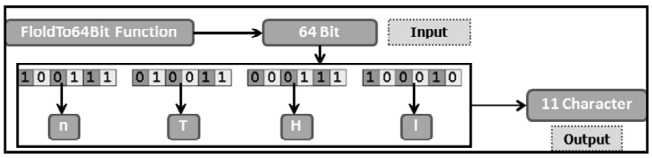
3. ที่มาและแรงจูงใจของปัญหา

การใช้งานรหัสผ่านซึ่งเป็น User Knowledge อย่างเดียวในการยืนยันตัวตนมีความเสี่ยงต่อระบบ เพราะรหัสผ่านรั่วไหลได้หลายช่องทาง จึงมีแนวคิดในการนำ OTP มาใช้ทำการยืนยันตัวตนด้วยปัจจัยสองแบบ (Two Factor Authentication:TFA) [2] สำหรับแก้ปัญหาที่เกิดขึ้น ความสำคัญนี้ระบบสารสนเทศชั้นนำของโลกหลายแห่งเลือกใช้ OTP เพื่อทำ TFA โดย OTP ที่เลือกเป็นแบบ HOTP และ TOTP แต่ OTP ทั้งสองแบบมีความเสี่ยงต่อการทำ Brute-force Attack สูง และถึงแม้จะมี Trial and Error Check เพื่อป้องกันแล้วก็ตาม แต่ด้วยความเสี่ยงที่มีมากต่อการ

Brute-force Attack ทำให้ค่า Trial and Error Check ของระบบมีค่าน้อย โดยปกติคือ 3 ครั้ง และหากผู้ใช้พิมพ์ผิดเกิน Trial and Error Check ที่ระบบตั้งไว้ ต้องมีภาระในการยืนยันความเป็นมนุษย์จริง หรือ CAPTCHA เพิ่มขึ้น ด้วยเหตุนี้องงานวิจัยนี้จึงต้องการลดความเสี่ยงต่อการทำ Brute-force Attack ของ HOTP และ TOTP เพื่อนำไปสู่การเพิ่มขึ้นของ Trial and Error Check ซึ่ง S/Key มีความสามารถในการลดความเสี่ยงดังกล่าว แต่ S/Key จะมีจำนวนตัวษรที่ต้องพิมพ์มากได้ถึง 24 ตัวอักษร ซึ่งอาจกระทบกับ Trial and Error Check ได้เช่นกัน งานวิจัยนี้จึงต้องการใช้ความสามารถของ S/Key เรื่องความเสี่ยงที่จะถูก Brute-force Attack ซึ่งน้อยกว่า HOPT และ TOTP มาปรับปรุงจำนวนตัวอักษรให้น้อยลง เพื่อทำให้การยืนยันตัวด้วย OTP แต่ละแบบมีประสิทธิภาพมากขึ้น

4. การออกแบบและพัฒนาระบบ

ออกแบบให้ใช้ BASE64 Data Encoding เพื่อลดจำนวนตัวอักษร- S/Key โดยเรียกว่า “BASE64 S/Key OTP” มีขั้นตอนดังแสดงในภาพที่ 1



ภาพที่ 1 *แสดงการลดจำนวนตัวอักษรของ S/Key ด้วย BASE64*

จาก FoldTo64Bit Function (อธิบายไว้ใน Appendix A ของ RFC 2289) ซึ่งเป็นฟังก์ชันลดจำนวน Bit จาก Hash Algorithm Function ของ S/Keyได้ Output 64 Bit ระบบใหม่นี้ใช้ BASE64 Data Encoding ซึ่งเป็นเทคนิคเพื่อแสดงผลข้อมูลจาก Bit ไปเป็นตัวอักษรและตัวเลข (A-Z, a-z, 0-9, +, /) ทั้งหมด 64 ตัว โดยใช้ 6 Bit ต่ออักษร 1 ตัว ดังแสดงในตารางที่ 2 ด้วยการนำ 64 Bit จาก Output ของ S/Key โดยแบ่ง Bit ทั้งหมดเป็น 11 ส่วนๆ ส่วนละ 6 Bit โดยส่วนที่ 11 ขาดข้อมูลไป 2 Bit เพิ่ม Bit ด้วย Even Parity Check Bit 2 ครั้ง นำแต่ละ ส่วนแปลงเป็นตัวอักษรจะได้ตัวอักษรทั้งหมด 11 ตัวอักษร เช่น 100111=n, 110110=2 เป็นต้น

จาก 64 Bit ซึ่งเป็น Output ของ FoldTo64Bit Function นั้นถูกรับรองจาก S/Key ว่าหาก Seed ที่นำเข้าไม่ว่าเป็น

ตารางที่ 2 *Bit to Character ของ BASE64 Data Encoding*

Binary	ASCII	Binary	ASCII	Binary	ASCII	Binary	ASCII
000000	A	010000	Q	100000	g	110000	w
000001	B	010001	R	100001	h	110001	x
000010	C	010010	S	100010	i	110010	y
000011	D	010011	T	100011	j	110011	z
000100	E	010100	U	100100	k	110100	0
000101	F	010101	V	100101	l	110101	1
000110	G	010110	W	100110	m	110110	2
000111	H	010111	X	100111	n	110111	3
001000	I	011000	Y	101000	o	111000	4
001001	J	011001	Z	101001	p	111001	5
001010	K	011010	a	101010	q	111010	6
001011	L	011011	b	101011	r	111011	7
001100	M	011100	c	101100	s	111100	8
001101	N	011101	d	101101	t	111101	9
001110	O	011110	e	101110	u	111110	+
001111	P	011111	f	101111	v	111111	/

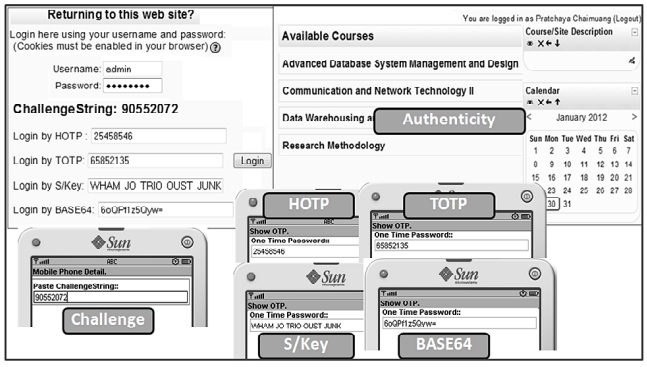
Counter ที่เพิ่มขึ้นตลอดเวลา, เวลาปัจจุบันที่เปลี่ยนไป หรือ Challenge ที่ได้จาก Server ถ้าไม่ซ้ำกันแล้วนั้น ผลของ 64 Bit ที่สร้างด้วย Algorithm ของ S/Key จะไม่ซ้ำกัน และจากการใช้ BASE64 S/Key OTP ที่ออกแบบใหม่นี้ เห็นได้ว่าข้อมูล Bit ทั้ง 64 Bit ที่ใช้ประมวลผล มาจากผลของ S/Key โดยตรง ทำให้ความหมายของข้อมูลที่เคยมีอยู่เช่น ความไม่ซ้ำกัน ซึ่งเป็นลักษณะเฉพาะของ OTP ไม่ได้ถูกเปลี่ยนแปลงไป จึงทำให้ข้อมูลแบบ BASE64 S/Key OTP ที่นำเสนอออกมาใหม่ไม่ได้เปลี่ยนแปลงความหมายใดๆ ของ S/Key แต่เปลี่ยนเพียงการนำเสนอข้อมูลที่มีจำนวนตัวอักษรน้อยลงเท่านั้น ซึ่งผลของการสร้าง BASE64 S/Key OTP จาก Seed แบบต่างแสดงตัวอย่างได้ดังตารางที่ 3

ตารางที่ 3 *ผลของ BASE64 S/Key OTP จาก Seed แต่ละแบบ*

Input Data		OTP Output	
type	seed	S/key OTP	BASE64 S/Key OTP
Counter	0	HOB SIT POW NET NUT NINA	G0eQ05XC2Wk=
	1	KUDO VARY BEND BELL COLD BOTH	mHxpWKr2nrs=
	2	AREA GIG HUT NASH KERR BEN	TeLcc1kJYg0=
	3	OVER MEN DRUB THEY DOOR DUKE	umUF3G03VvA=
	4	MAP MOE QUIT OLGA WING REAL	J0Um69v+w3o=
Time	1970-01-01 00:00:59	SOAR DOSE SOAK SEA GAL AQUA	0S63RB2BXJs=
	2011-12-20 01:58:29	WANE ONTO MULE TOTE VINE BOLO	5xcewW8+ULY=
	2011-12-20 01:58:30	PAN MUTE NICE NODE GORY MY	MfYe0Np48FQ=
	2011-12-20 01:58:31	BIG ARMY EDGY YALE REEF	B57tOT0+6Xs=
	2011-12-20 01:58:32	BOMB ROOF GAD NAIR ALIA	W7g4VljUr+4=
Challenge	90552072	WHAM JO TRIO OUST JUNK KICK	6oQPf1z5Qyw=
	32546588	BURL OVA MAO WALL ACTA	YKYXyROeapE=
	12654858	LETS DING EEL SELL LETS	9XO1zYhskzs=
	56854254	WAVY FOWL BLUM CRAM RIFT	6IC72q72w3R=
	75568751	GLAD JUG GIRD JUNK SHOW	jEQmL33JQ5Y=

จากนี้ในงานวิจัยนี้ยังได้พัฒนาโปรแกรมต้นแบบเพื่อทดสอบการทำงานของ BASE64 S/Key OTP ด้วยการนำ User Possession [1] แบบต่างๆ มาแสดงผล OTP เพื่อทำ TFA

โดยใช้ข้อมูล Login สองส่วน คือ รหัสผ่าน จาก User Knowledge และ OTP จาก User Possession ซึ่งวิธีการนี้เป็นที่นิยมใช้งานมากที่สุดในปัจจุบันโดยในงานวิจัยนี้ได้เลือกโทรศัพท์มือถือที่ผู้ใช้พกพาอยู่แล้ว นำมาเป็น User Possession ที่ใช้แสดง OTP เพื่อไม่เกิดค่าจ่ายในการซื้อ User Possession แบบอื่นๆ ด้วยการพัฒนาโปรแกรมต้นแบบในรูปของ PHP API และโปรแกรม J2ME ที่สามารถสร้างตรวจสอบ OTP ได้ 4 แบบ คือ HOTP, TOTP, S/Key และBASE64 S/Key OTP โดยเลือกใช้ Seed แบบ Challenge ที่สร้างโดย Server และแสดงที่หน้า Login ซึ่ง Challenge สร้างด้วยการสุ่มตาม RFC 4086 [12] ซึ่งเป็นมาฐานการสุ่มข้อมูลที่ IETF แนะนำให้เป็น "BEST CURRENT PRACTICE" (การทำงานที่ดีที่สุดในปัจจุบัน) ในการสุ่มข้อมูล Challenge ที่ไม่ซ้ำกันผลการพัฒนาโปรแกรมมีดังต่อไปนี้



ภาพที่ 2 *ผลการพัฒนาโปรแกรมต้นแบบ*

จากภาพที่ 2 แสดงผลการพัฒนาโปรแกรมโดยนำไปใช้กับ MOODEL [13] ซึ่งเป็น E-Learning หนึ่งในรายวิชาที่เปิดสอนอยู่ในรายวิชาเครือข่ายคอมพิวเตอร์1 สาขาเทคโนโลยีสารสนเทศ คณะวิทยาการสารสนเทศ มหาวิทยาลัยมหาสารคาม โดยมีผู้ใช้ทั้งหมดจำนวน 115 คน เพื่อแสดงถึงการนำไปใช้งานกับระบบจริง ทำโดยด้วยการเปลี่ยนกระบวนการ Login ของ MOODEL ใหม่ให้สามารถ Login ได้ด้วย OTP แต่ละแบบ ซึ่งที่หน้า Login จะแสดง Challenge String เพื่อใช้เป็น Seed ในการสร้าง OTP พร้อมกับพัฒนาโปรแกรม J2ME เพื่อติดตั้งที่โทรศัพท์มือถือของผู้ใช้ ซึ่งจะนำไปสร้าง และแสดงผล OTP แบบต่างๆ โดยการรับ Seed (Challenge String) จากหน้า Login และพิมพ์ Seed ที่เห็นลงที่โปรแกรมบนมือถือ พร้อมกับเลือกตัวเลือกในการแสดงผลซึ่งจะแสดงผลได้ 4 แบบ คือ HOTP, TOTP, S/Key และ BASE64 S/Key OTP และจึงส่ง OTP ที่ได้แบบใดแบบหนึ่งให้ Server เพื่อตรวจสอบตามฟอร์มการรับข้อมูลจากหน้า Login และเมื่อ Server ได้รับข้อมูล OTP ในแบบต่างๆ ก็จะทำการสร้าง OTP นั้นตาม Algorithm นั้นๆ โดยใช้ Seed ตัวเดียวกันกับที่แสดงต่อผู้ใช้ หาก OTP ตรงกันแสดงว่าการยืนยันตัวตนถูกต้อง

จากการพัฒนาระบบในส่วนนี้เห็นได้ว่าได้ออกแบบให้มีการยืนยันตัวด้วย OTP ถึง 4 แบบ รวมไปถึง BASE64 S/Key OTP ซึ่งเป็น OTP ใหม่ ที่ได้พัฒนาขึ้น เพื่อเก็บสถิติในการ Login ด้วย OTP แต่ละแบบและเก็บสถิติการพิมพ์ข้อมูลผิดพลาด ซึ่งเกิดการใช้ OTP ที่มีความยาวและลักษณะตัวอักษรแตกต่างกัน ผลที่ได้เป็นดังต่อไปนี้

HTOP Login	HOTP Error	TOTP Login	TOTP Error	Skey Login	Skey Error	Base64 Login	Base64 Error
446	36	653	46	376	105	458	61

ภาพที่ 3 *แสดงจำนวนการ Login ด้วย OTP แต่ละแบบ*

จากภาพที่ 3 แสดงการเก็บสถิติการ Login ด้วย OTP แต่ละแบบใน MOODEL ที่ใช้ทดสอบเป็นเวลา 2 เดือน (1 กันยายน - 30 พฤศจิกายน 2554) จากผู้ใช้ทั้งหมด 115 คน ผลที่ได้ คือ

HOTP Login 466 ครั้ง พิมพ์ผิด 36 ครั้ง คิดเป็น 8%
TOTP Login 653 ครั้ง พิมพ์ผิด 46 ครั้ง คิดเป็น 7%
S/Key Login 376 ครั้ง พิมพ์ผิด 192 ครั้ง คิดเป็น 51%
Base64 Login 458 ครั้ง พิมพ์ผิด 61 ครั้งคิดเป็น 13%

5. ผลการทดสอบและวิเคราะห์เปรียบเทียบ

5.1 การวิเคราะห์ผลของ OTP แต่ละแบบ ในส่วนนี้เป็นการแสดงให้เห็นถึงความเป็นไปได้ ที่จะเกิดผลของ OTP แต่ละแบบ รวมไปถึง OTP ที่ออกแบบใหม่ คือ BASE64 S/Key OTP เพื่อวิเคราะห์ความเสี่ยงจาก Brute-force Attack ของ OTP แบบใหม่เมื่อเทียบกับ OTP แบบเดิม

ตารางที่ 4 *ความเป็นไปได้ที่จะเกิดผลทั้งหมดของ OTP แต่ละแบบ*

OTP Type	Choice	Output	Probability (P)	Log10(P)	
HOTP/TOTP	10	6	10,000,000	7	10^7
HOTP/TOTP	10	7	100,000,000	8	10^8
HOTP/TOTP	10	8	1,000,000,000	9	10^9
HOTP/TOTP	10	19	10,000,000,000,000,000,000	19	10^19
S/Key	2048	6 [4-24]	73,786,976,294,838,206,464	19.86798	10^19
BASE64	64	11	73,786,976,294,838,206,464	19.86798	10^19

จากตารางที่ 4 เห็นได้ว่าความเป็นได้ที่จะเกิดผลทั้งหมดของ BASE64 S/Key OTP คือ $P_{(BASE64\ S/Key\ OTP)} = 64^{11}$ (มาจากความเป็นไปได้ที่จะเกิด BASE64 S/Key OTP หนึ่งหลักคือ 64 ตามตารางที่ 2 ใช้ทั้งหมด 11 หลัก) หรือประมาณ 10^{19} จากข้อมูลข้างต้นจะเห็นได้ว่า $P_{(BASE64\ S/Key\ OTP)} = P_{(S/Key)}$ แต่ BASE64

S/Key OTP แสดงผลตัวอักษรในจำนวนที่น้อยกว่า คือ เพียง 11 ตัวอักษรเท่านั้น และนอกจากนี้ค่า P ของ BASE64 S/Key OTP ยังมากกว่า HOTP และ TOTP ซึ่งหากจะทำให้ค่า P ของ HOTP และ TOTP ใกล้เคียงกับ BASE64 S/Key OTP ต้องเพิ่มจำนวนตัวอักษร Output ของ OTP ทั้งสองแบบไปถึง 19 ตัวอักษรดังตารางที่ 4 และจากความเป็นไปได้ที่จะเกิด OTP ทั้งหมดที่มีมาก ทำให้การ Brute-force Attack ทำได้ยากขึ้นด้วยเหตุเองทำให้ BASE64 S/Key OTP มีความสามารถการป้องกันการ Brute-force Attack ได้เทียบเท่ากับ S/Key แต่มีจำนวน Output ของ OTP น้อยกว่า และยังปลอดภัยกว่า HOTP และ TOTP

5.2 **การวิเคราะห์ Trial and Error Check** จากผลการทดสอบจากการใช้งานกับระบบจริงเห็นได้ว่า BASE64 S/Key OTP ถูกเลือกใช้ Login ใกล้เคียงกับ HOTP และ TOTP และมากกว่า S/Key ชี้ให้เห็นว่าการพิมพ์ตัวอักษร 11 ตัว ซึ่งอยู่ในรูปแบบของ BASE64 S/Key OTP จากผู้ใช้โดยทั่วไปนั้นยอมรับได้ หากดูจากจำนวนการเลือกที่จะใช้ OTP ดังกล่าว Login แต่การพิมพ์ข้อมูลผิดยังสูงกว่า HOTP และ TOTP ซึ่งอาจจะเนื่องจากจำนวนตัวอักษรที่พิมพ์มากกว่า แต่เมื่อหากเปรียบเทียบจากความสามารถในการป้องกัน Brute-force Attack นั้น BASE64 S/Key OTP มีความสามารถสูงกว่าอย่างมหาศาล ซึ่งจากผลดังกล่าวหากนำ BASE64 S/Key OTP ไปใช้งานแทน HOTP และ TOTP นั้น จะสามารถเพิ่ม Trial and Error Check ที่จากเดิมกำหนดไว้ที่ 3 ครั้งให้มากขึ้นได้เนื่องจากมีความเสี่ยงจาก Brute-force Attack น้อยกว่ามาก จะทำให้ผู้ใช้สามารถพิมพ์ข้อมูล Login ผิดพลาดได้มากขึ้น และลดกระบวนการตรวจสอบ CAPTCHA อันเนื่องมาจากการกรอกข้อมูล Login ผิดพลาดเกินค่า Trial and Error Check ลงได้ และทำให้ตัวระบบเองยังมีความปลอดภัยจากการทำ Brute-force Attack เพิ่มมากขึ้นหลายเท่าตัว

6. สรุปผลการพัฒนา

งานวิจัยนี้แสดงให้เห็นถึงการปรับปรุง OTP แบบ S/Key เรียกว่า BASE64 S/Key OTP ด้วยการใช้เทคนิค BASE64 Data Encoding เพื่อให้แสดงผลของ OTP ใหม่ที่มีจำนวนตัวอักษรน้อยกว่าเดิม แต่ยังปลอดภัยจากการทำ Brute-force Attack เท่าเดิม และปลอดภัยกว่า OTP แบบ HOTP และ TOPT ซึ่งเป็น OTP แบบที่นิยมใช้งานอย่างแพร่หลายในปัจจุบัน ทำให้สามารถแก้ความเสี่ยงจากการทำ Brute-force Attack ของ HOTP และ TOTP ได้อย่างมีประสิทธิภาพ และยังสามารถเพิ่ม Trial and Error Check ให้กับระบบต่างๆ ที่ใช้ OTP ซึ่งเป็นการเพิ่มจำนวนครั้งในการพิมพ์ OTP ทำให้ลดกระบวนการที่ต้องตรวจสอบหากพิมพ์ข้อมูล Login ผิดให้กับผู้ใช้ได้ และระบบยังปลอดภัยจากการทำ Brute-force Attack เพิ่มมากขึ้น

7. ข้อเสนอแนะในงานวิจัย

เนื่องจากระบบการยืนยันตัวตนบนระบบสารสนเทศส่วนใหญ่ยังคงใช้รหัสผ่าน ซึ่งเป็น User Knowledge เช่นเดียวกันกับงานวิจัยนี้ยังใช้ User Knowledge เป็นปัจจัยส่วนหนึ่งในกระบวนการดังกล่าว ซึ่ง User Knowledge นั้นมีอีกปัญหาหนึ่งที่พบ คือ ปัญหาการลืม และถึงแม้ว่าปัญหานี้ได้มีวิธีแก้ไขแล้วในปัจจุบัน เช่น การกู้คืนรหัสผ่านโดยใช้คำถามกันลืม หรือใช้ E-Mail เพื่อส่งรหัสผ่านใหม่ในกรณีลืม เป็นต้น แต่วิธีที่นำเสนอดังกล่าวก็ยังเกิดปัญหา เช่น ลืมคำกันลืม, ระบบ E-Mail ไม่ปลอดภัย ดังนั้นระบบที่ยังต้องรหัสผ่านเป็นปัจจัยหลักในการยืนยันยันตัวตน จำเป็นต้องมีการกู้คืนรหัสผ่านที่เหมาะสมและปลอดภัย ด้วยการนำปัจจัยตัวตนประเภทอื่นเข้ามาประยุกต์ใช้งานเช่นใช้ User Attribute [1] ที่ไม่สามารถถูกลืม หรือ รั่วไหลได้ เป็นต้น

8. เอกสารอ้างอิง

[1] T. Plum and R. Bleiler, *User authentication*, SPEC Kits: Washington D.C, 2001.

[2] S. Garfinkel, G. Spafford, and A. Schwartz, *Practical UNIX and Internet Security*, O'Reilly Media: California, 2003.

[3] D. Raihi, M. Bellare, F. Hoornaert, D. Naccache, and O. Ranen, "HOTP: An HMAC-Based One-Time Password Algorithm," *IETF,* RFC 4226, December 2005.

[4] D. Raihi, S. Machani, M. Pei, and J. Rydell, "TOTP: Time-Based One-Time Password Algorithm," *IETF,* RFC 6238, May 2011.

[5] N. Haller, C. Metz, P. Nesser, and M. Straw, "A One-Time Password System," *IETF,* RFC 2289, February 1998.

[6] M. Curtin, *Brute force: cracking the data encryption standard*, Springer: New York, 2005.

[7] H. Nahari and Ronald L Krutz, *Web Commerce Security: Design and Development*, Wiley Publishing, Inc: Indianapolis, 2011.

[8] L. Kang and J. Xiang, "CAPTCHA Phishing: A Practical Attack on Human Interaction Proofing," in *Information Security and Cryptology: 5th International Conference, Inscrypt 2009*, Beijing, China, pp. 411-425, 12-15 December, 2009.

[9] S. Josefsson, "The Base16 Base32 and Base64 Data Encodings," *IETF,* RFC 4648, October, 2006.

[10] H. Krawczyk, M. Bellare, and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication," *IETF,* RFC 2104, February 1997.

[11] Password Recovery Speeds, Available online at http://www.lockdown.co.uk/?pg=combi&s=articles.

[12] D. Eastlake, J. Schiller, and S. Crocker, "Randomness Requirements for Security", *IETF,* RFC 4086, June 2005.

[13] Moodle, Available online at http://moodle.org.