บทความวิจัย

การประชุมวิชาการ งานวิจัยและพัฒนาเชิงประยุกต์ ครั้งที่ 8

8th ECTI-CARD 2016, Hua Hin, Thailand

ระบบยืนยันตัวตนแบบ 2 ขั้นตอนอย่างปลอดภัยสำหรับใช้งานทั่วไป

2-Step Verification Secure System for General Use

ประภาส ผ่องสนาม, อรอนงค์ แซ่จิว

สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษมบัณฑิต

60 ถนนร่มเกล้า แขวงมีนบุรี เขตมีนบุรี กรุงเทพฯ 10510

E-mail: p.phongsanam@gmail.com

บทคัดย่อ

ในสภาพแวดล้อมของการเข้าใช้งานอินเตอร์เน็ตและการสื่อสารไร้สายนั้น การยืนยันตัวตนถือเป็นสิ่งที่สำคัญมากในโลกยุคปัจจุบัน ดังนั้นในบทความนี้นำเสนอกระบวนการปรับปรุงระบบยืนยันตัวตนรูปแบบเดิมให้มีความปลอดภัยมากยิ่งขึ้น ในการเข้าถึงระบบสารสนเทศที่ใช้งานอยู่ทั่วไป โดยใช้การยืนยันตัวตนแบบ 2 ขั้นตอน (2-Step Verification) โดยประยุกต์ใช้แอปพลิเคชันที่ติดตั้งบนสมาร์ทโฟน ซึ่งเป็นวิธีการที่ง่าย สะดวกใช้งาน และผู้ใช้งานทั่วไปสามารถนำไปประยุกต์ใช้ได้จริงในชีวิตประจำวัน ซึ่งจากผลการศึกษาและทดลองพบว่า การเพิ่มกระบวนการยืนยันตัวตนแบบ 2 ขั้นตอนนั้นสามารถเพิ่มความปลอดภัยให้กับผู้ใช้งานระบบและลดความซับซ้อนในการกำหนดรหัสผ่านของผู้ใช้งานได้เป็นอย่างดี

คำสำคัญ: การยืนยันตัวตนแบบ 2 ขั้นตอน, ระบบล็อกอิน, ระบบยืนยันตัวตน

Abstract

In the Internet and Wireless communication environment, authentication of the users is very important. This article proposed scheme to improve security in a general authentication system by using 2-steps verification from smartphone application. The propose scheme is simple, flexible and can used in daily life. The simulation and results shown adding 2-steps, verification process can increase the security of the system and simplify the password of users.

Keywords: 2-Step Verification, login system, authentication system

1. บทนำ

การใช้งานอีเมล์และระบบสารสนเทศในปัจจุบันแทบจะเป็นปัจจัยสำคัญปัจจัยหนึ่งสำหรับโลกในยุคปัจจุบัน ที่ทุกคนสามารถเข้าถึงอินเตอร์เน็ตและเทคโนโลยีสมัยใหม่ได้ผ่านทางมือถือและแอปพลิเคชันต่าง ๆ ซึ่งหากผู้ใช้งานไม่ตระหนักถึงความเสี่ยง หรือภัยคุกคามต่าง ๆ ที่อาจเกิดขึ้นได้จากความรู้เท่าไม่ถึงการณ์ก็อาจนำไปสู่ความเสียหายที่อาจประเมินค่าไม่ได้ ณ ปัจจุบันการยืนยันตัวตนเพื่อเข้างานระบบสารสนเทศต่าง ๆ ยังคงใช้การยืนยันตัวตนในรูปแบบเดิมคือใช้ชื่อผู้ใช้งาน (Username) และรหัสผ่าน(Password) หากผู้ใช้งานกำหนดชื่อผู้ใช้งานหรือรหัสผ่านที่ง่ายเกินไป[1-3] เช่น 123456, password, 12345, 12345678, qwerty, วัน เดือน ปีเกิด ฯลฯ ก็อาจถูกผู้ไม่ประสงค์ดีสามารถคาดเดารหัสผ่านเพื่อนำไปใช้ยืนยันตัวตนเข้าสู่ระบบเพื่อเข้าไปดำเนินการธุรกรรมต่าง ๆ แทนผู้ใช้งานจริงได้ซึ่งอาจเกิดความเสียหายต่าง ๆ ตามมา ซึ่งในการล็อกอินเข้าใช้งานระบบโดยใช้ชื่อผู้ใช้งาน และรหัสผ่าน นอกจากจะต้องทำงานผ่านช่องทางที่ปลอดภัย (Secure Channel) หรือ SSL (Secure Sockets Layer) เพื่อป้องกันแฮคเกอร์ (Hacker) หรือผู้ไม่ประสงค์ดีดักจับข้อมูลระหว่างทางแล้ว ปัจจัยหนึ่งที่ต้องคำนึงถึงในด้านความปลอดภัยคือการกำหนดชื่อผู้ใช้งานและรหัสผ่านต้องมีความปลอดภัยและยากต่อการคาดเดาด้วย ซึ่งการกำหนดรหัสผ่านเพื่อความปลอดภัยในการยากที่จะคาดเดานั้นจะขึ้นอยู่กับความยาวตัวอักษร[4][5] และต้องประกอบด้วยตัวอักษรตัวเล็ก ตัวใหญ่ ตัวเลข อักษรพิเศษและต้องมีการเปลี่ยนรหัสผ่านใหม่เรื่อย ๆ ด้วย ซึ่งการกำหนดรหัสผ่านที่ยาว เพื่อความปลอดภัยนั้นจะนำไปสู่ปัญหาการลืมรหัสผ่านหรือการพิมพ์รหัสผ่านผิดเพิ่มขึ้นด้วย ดังนั้นในระบบที่ต้องการความปลอดภัยสูงเช่นระบบธนาคารจะมีการใช้รหัส OTP (One Time Password) โดยมีการส่งรหัสลับไปยังมือถือของผู้ใช้งานในแต่ละครั้งที่มีการทำรายการซึ่งเป็นวิธีการที่สะดวกและปลอดภัย แต่ก็ต้องแลกมากับการเสียค่าใช้จ่ายที่เพิ่มมากขึ้นเพื่อแลกกับความปลอดภัยที่เพิ่มเข้ามา ซึ่งการเพิ่มค่าใช้จ่ายอาจไม่เหมาะสมกับธุรกิจขนาดเล็กที่มีต้นทุนน้อย

ในปัจจุบันแทบทุกคนจะมีการใช้งานโทรศัพท์มือถือแบบสมาร์ทโฟน ซึ่งสมาร์ทโฟนส่วนใหญ่นั้นจะสามารถติดตั้งโปรแกรมเพื่ออำนวยความสะดวกในด้านต่าง ๆ เช่น ด้านบันเทิง ข่าวสาร กีฬา สุขภาพ ตลอดถึง แอปพลิเคชันด้านการเงิน การธนาคาร และด้านความปลอดภัยต่าง ๆ ซึ่ง ณ ปัจจุบันได้มีแอปพลิเคชันเพื่อการสร้างรหัสลับอย่างปลอดภัยที่มีการเปลี่ยนแปลงเลขรหัสตามช่วงเวลาเพื่อใช้เป็นรหัสด้าน

ความปลอดภัย ซึ่งในบทความนี้ได้นำเสนอการปรับปรุงระบบรูปแบบเดิมเพื่อให้สามารถประยุกต์ใช้งานร่วมกับแอปพลิเคชันสร้างรหัสลับตามช่วงเวลา เพื่อให้กระบวนการยืนยันตัวตนในรูปแบบเดิมนั้นมีความปลอดภัยเพิ่มมากยิ่งขึ้นและลดปัญหาความซับซ้อนในการกำหนดรหัสผ่านสำหรับผู้ใช้งานได้ด้วย

2. ทฤษฎีพื้นฐาน

Time-based One-time Password Algorithm (TOTP) : คือกระบวนการที่ใช้คอมพิวเตอร์สร้างรหัสเฉพาะตามแต่ละช่วงเวลาที่กำหนดไว้จากแชร์คีย์ลับ (Shared secret key) ที่สร้างขึ้นตามมาตรฐาน RFC 6238 [6][7] ซึ่งในเวลาต่อมาได้นำเอาไปใช้เป็นพื้นฐานสำหรับระบบระบบ Open Authentication (OATH) และ มีการนำหลักการนี้ไปใช้ในการสร้างตัวเลขในกระบวนการยืนยันตัวตน แบบ 2 ปัจจัย (Two Factor Authentication) โดยในการสร้างรหัสตามช่วงเวลานั้นจะมีการนำค่าคีย์ลับ(secret key) มารวมกับช่วงเวลาประทับ (timestamp) ผ่านฟังก์ชันเข้ารหัสแฮช (cryptographic hash function) เพื่อสร้างเป็นรหัสตามแต่ละช่วงเวลา ซึ่งโดยปกติแล้วค่าเวลาประทับจะถูกกำหนดให้มีการเปลี่ยนแปลงไปในทุก ๆ 30 วินาที ซึ่งทำให้รหัสที่ได้จะเปลี่ยนแปลงไปเรื่อย ๆ ตามช่วงเวลา โดยในแต่ละช่วงเวลารหัสที่ถูกสร้างขึ้นมาจากคีย์ลับตัวเดียวกันจะต้องมีค่าผลลัพธ์เมื่อผ่านฟังก์ชันแฮชแล้วเป็นค่าเดียวกันเสมอ

Google Authenticator (GA) [8-10] : คือโปรแกรมประยุกต์ที่สามารถดาวน์โหลดและติดตั้งได้ฟรีทั้งบนสมาร์ทโฟนหรือคอมพิวเตอร์ โดยโปรแกรม GA จะทำหน้าที่สร้างรหัส TOTP ตามมาตรฐาน RFC 6238 โดยรหัสที่โปรแกรม GA สร้างมาให้นั้นจะเป็นตัวเลขที่มีขนาดความยาวตั้งแต่ 6 – 8 หลักซึ่งกระบวนการสร้างรหัสโดยโปรแกรม GA ตามช่วงเวลา (Time Based) แสดงดังรูปที่ 1

```
function GoogleAuthenticatorCode(string secret)
  key := base32decode(secret)
  message := floor(current Unix time / 30)
  hash := HMAC-SHA1(key, message)
  offset := last nibble of hash
   /*4 bytes starting at the offset*/
  truncatedHash := hash[offset..offset+3]
   /*remove  most significant bit*/
  Set the first bit of truncatedHash to zero
  code := truncatedHash mod 1000000
  pad code with 0 until length of code is 6
  return code
```

รูปที่ 1 Pseudocode for One Time Password OTP

3. การออกแบบและนำไปใช้งาน

การปรับปรุงระบบยืนยันตัวตนรูปแบบเดิมโดยเพิ่มขั้นตอนการเก็บบันทึกและตรวจสอบรหัส TOTP นั้นเราได้ทำการเพิ่มฟิลด์ GA_key และ GA_enable เข้าไปในโครงสร้างตารางแสดงดังรูปที่ 2 เพื่อใช้สำหรับเก็บรหัสลับและสถานะการเปิดใช้งาน TOTP ตามลำดับ

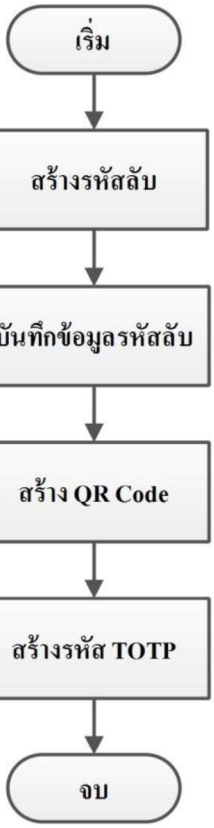
```
CREATE TABLE  user_password (
        username VARCHAR( 32 ) NOT NULL ,
        password VARCHAR( 32 ) NOT NULL ,
        GA_key CHAR( 16 ) NOT NULL ,
        GA_enable TINYINT( 1 ) NOT NULL ,
        PRIMARY KEY ( `username` )
)
```

รูปที่ 2 โครงสร้างตารางสำหรับเก็บข้อมูลชื่อผู้ใช้งาน รหัสผ่าน

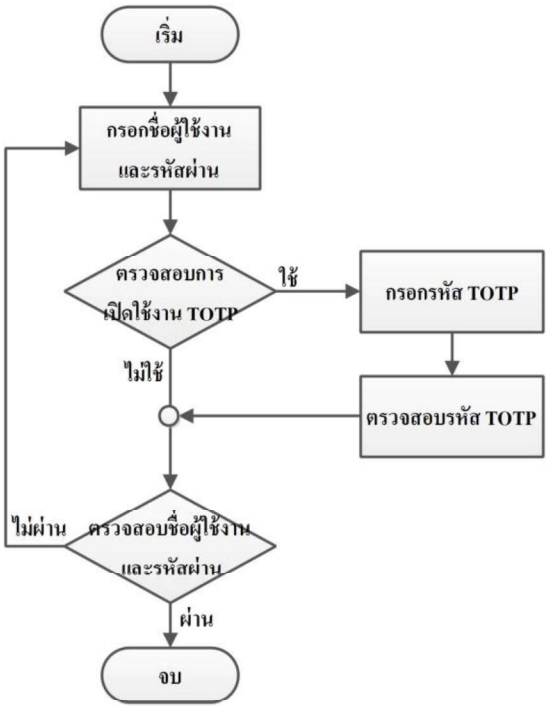
โดยในกระบวนการเก็บบันทึกรหัสลับและกระบวนการยืนยันตัวตนจะมีกระบวนการทำงานดังรูปที่ 3 -4 ตามลำดับ

4. ผลการทดลอง

ในขั้นตอนการทดสอบเราได้ทำการเขียนโปรแกรมด้วยภาษา PHP [9] เพื่อทำการสร้างรหัสลับ (Secret key) ขนาดความยาว 16 ตัวอักษรพร้อมทั้ง QR-Code เพื่อใช้เป็นชุดข้อมูลเริ่มต้นในการสร้างรหัส TOTP สำหรับใช้กับโปรแกรม GA แสดงดังรูปที่ 5 ซึ่งผู้ใช้งานสามารถเลือกกรอกรหัสตัวอักษรเข้าไปในโปรแกรม GA หรือจะใช้วิธีการแสกน QR-Code ที่โปรแกรมสร้างให้ก็ได้ จากตัวอย่างรหัสลับคือ “**APJZW4PN7NMGS3ZT**” ซึ่งเมื่อนำรหัสนี้ไปกรอกในโปรแกรม GA แสดงดังรูปที่ 6 -7 ตามลำดับ ซึ่งสามารถทำได้ 2 วิธีคือแสกน QR-Code และกรอกเลขรหัสเข้าไปด้วยตัวเอง โดยเมื่อกรอกรหัสเข้าไปในโปรแกรม GA แล้วจะต้องได้รหัส TOTP หรือ GA Code เป็น “**096843**” แสดงดังรูปที่ 8 ตามลำดับ ซึ่งเลขรหัสนี้จะเปลี่ยนแปลงไปทุก ๆ 30 วินาที ซึ่งรหัสนี้จะนำไปใช้เป็นรหัสเพื่อใช้สำหรับยืนยันตัวตนในขั้นตอนที่ 2 หลังจากที่ผู้ใช้งานได้ล็อกอินด้วยชื่อผู้ใช้งาน และ รหัสผ่านจากขั้นตอนเดิมแล้ว

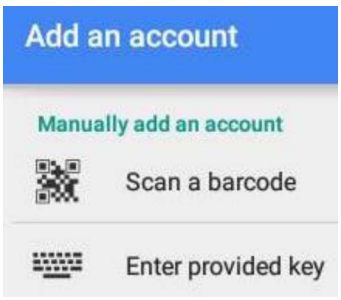


รูปที่ 3 แสดงขั้นตอนการสร้าง TOTP



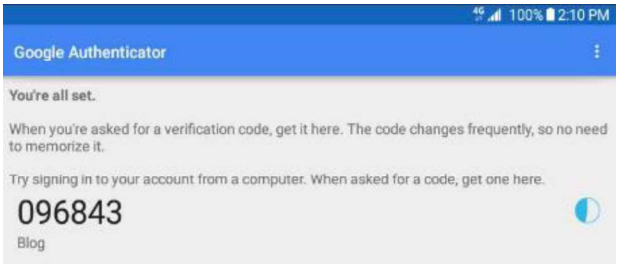
รูปที่ 4 แสดงกระบวนการ ตรวจสอบเพื่อยืนยันตัวตน

รูปที่ 5 ตัวอย่างโปรแกรมสร้างรหัสลับและ QR-Code



รูปที่ 6 แสดงหน้าต่างนำเข้าข้อมูลรหัสลับเพื่อใช้สร้างรหัส TOTP

รูปที่ 7 แสดงหน้าต่างกรอกรหัส



รูปที่ 8 แสดงรหัส TOTP ที่ได้จากโปรแกรม GA

5. สรุป

การนำรหัส TOTP มาใช้เพื่อยืนยันตัวตนแบบ 2 ขั้นตอนนั้น สามารถเพิ่มความปลอดภัยให้กับผู้ใช้งานและสามารถลดความซับซ้อนในการกำหนดรหัสผ่านของผู้ใช้งานได้ อีกทั้งโปรแกรม GA สามารถดาวน์โหลดและติดตั้งลงบนเครื่องคอมพิวเตอร์หรือสมาร์ทโฟนเพื่อใช้งานได้ฟรีซึ่งเป็นวิธีการที่สะดวกและเพิ่มความปลอดภัยของระบบได้มากยิ่งขึ้น

เอกสารอ้างอิง

[1] SplashData, Inc., “Worst Passwords List”, http://splashdata.com/blog/

[2] J. Condliffe, “The 25 Most Popular Passwords of 2014”, http://gizmodo.com/the-25-most-popular-passwords-of-2014-were-all-doomed-1680596951

[3] M. Slain, “Announcing Our Worst Passwords of 2015”, https://www.teamsid.com/worst-passwords-2015/

[4] M. McDowell, S. Hernan, J. Rafail, “Cyber Security Tip ST04-002, Choosing and Protecting Passwords,” US CERT. Retrieved June 20, 2009.

[5] “Want to deter hackers? Make your password longer”. MSNBC. 2010-08-19. Retrieved 2010-11-07.

[6] “RFC 6238 - TOTP: Time-Based One-Time Password Algorithm”. Retrieved July 13, 2011.

[7] RFC 6238, https://tools.ietf.org/html/rfc6238

[8] Google Authenticator, https://en.wikipedia.org/wiki/Google_Authenticator

[9] PHPGangsta/GoogleAuthenticator, https://github.com/PHPGangsta/GoogleAuthenticator

[10] chregu/GoogleAuthenticator, https://github.com/chregu/GoogleAuthenticator.php

[11] Hypertext Preprocessor, http://php.net/