การประเมินวิธีแก้ไขปัญหาการโจมตีด้วยการเปลือยเอสเอสแอล
Experimental Evaluation of SSL Stripping Attack Solutions

สมนึก พ่วงพรพิทักษ์ (Somnuk Puangpronpitag)* และ อภิรักษ์ ทูลธรรม (Apirak Toolthamm)**

บทคัดย่อ

เมื่อไม่นานมานี้ การโจมตีด้วยเทคนิคการเปลือยเอสเอสแอล (SSL Stripping) ถูกแฮ็กเกอร์จำนวนมากนำไปใช้เพื่อทำลายการรักษาความมั่นคงของระบบธนาคารออนไลน์ เว็บไซต์พาณิชย์อิเล็กทรอนิกส์ (E-commerce) และระบบโปรแกรมประยุกต์เว็บอื่นๆ ทั้งนี้ ได้มีงานวิจัยหลายชิ้นที่ได้ศึกษาถึงเทคนิคการโจมตีและนำเสนอวิธีการแก้ปัญหา แต่การแก้ปัญหาเหล่านั้นล้วนไม่ได้รับการตรวจสอบอย่างจริงจัง และยังมีคำถามว่าสามารถปกป้องระบบได้จริงหรือไม่ งานวิจัยนี้มีจุดมุ่งหมายเพื่อประเมินวิธีการแก้ไขปัญหาที่ถูกนำเสนอเหล่านั้น ด้วยการทดสอบบนเครือข่ายทดลอง (Test-bed) โดยใช้ประสิทธิผลของการป้องกันความเป็นมิตรกับผู้ใช้ข้อจำกัดในการรับมือการโจมตีของ Script Kiddies ได้เท่านั้น และความครอบคลุมการทำงานของแพลตฟอร์ม เป็นเกณฑ์ในการประเมินจากการทดสอบเปิดเผยให้เห็นว่าวิธีการแก้ไขปัญญาที่ถูกนำเสนอเหล่านั้นสามารถถูกทำลายได้อย่างง่ายดายด้วยการแก้ไขโค้ดของการโจมตีเพียงไม่กี่บรรทัดของแฮ็กเกอร์ที่มีความเชี่ยวชาญ ซึ่งผลจากงานวิจัยนี้จะสามารถนำไปใช้เป็นแนวทางการออกแบบวิธีการแก้ปัญหารูปแบบใหม่ที่มีประสิทธิภาพในการป้องกันการโจมตีจากเทคนิค SSL Stripping ต่อไป

คำสำคัญ: การโจมตีด้วยการเปลือยเอสเอสแอล การโจมตีเอชทีทีพีเอส การโจมตีแบบแทรกกลางการสื่อสาร การโจมตีโปรแกรมประยุกต์เว็บ

Abstract

Recently, SSL stripping attacks have been deployed by several hackers to break the security of online banking services, e-commerce webs and other web applications. Several studies have done to investigate the attacking techniques, and several solutions have also been proposed. However, the proposed solutions have not been investigated seriously, and still very questionable. This paper aims at evaluating extensively the previous proposed solutions using an experimental test-bed. Effectiveness, user-friendliness, script-kiddy-only limitation, platform coverage are our proposed evaluation criteria. The experimental results from this work can direct to the design of a new effective solution for the SSL stripping attacks.

Keyword: SSL Stripping Attack, HTTPS Attack, Man in the Middle Attack, Web Application Attack.

1. บทนำ

การโจมตีด้วยการเปลือยเอสเอสแอล (SSL Stripping Attack) ถูกใช้อย่างแพร่หลายในการจู่โจมระบบให้บริการของโปรแกรมประยุกต์เว็บ (Web Application) เช่น Online Banking, E-Commerce, Webmail และ Social Network ที่ปัจจุบันแม้จะมี Hypertext Transfer Protocol Secure (HTTPS) ซึ่งใช้ Secure Socket Layer (SSL) เข้ารหัสการสื่อสารและยืนยันตัวตนระหว่างไคลเอนต์และเซิร์ฟเวอร์ก็ตาม การจู่โจมด้วยเทคนิคดังกล่าวก็ยังสามารถโจมตีเพื่อดักจับข้อมูลสำคัญอย่างชื่อผู้ใช้และรหัสผ่านออกมาได้ ซึ่งเป็นภัยคุกคามที่สร้างความสูญเสียร้ายแรงให้กับผู้ใช้

ในช่วงระยะเวลาที่ผ่านตั้งแต่ปี ค.ศ. 2009 จนถึงปัจจุบัน (เมษายน ค.ศ. 2013) เกิดคดีอาชญากรรมทางคอมพิวเตอร์และความเสียหายทางเศรษฐกิจเป็นจำนวนมากบนระบบ

* สาขาวิชาวิทยาการคอมพิวเตอร์ คณะวิทยาการสารสนเทศ มหาวิทยาลัยมหาสารคาม
** สาขาวิชาบริหารธุรกิจ คณะอุตสาหกรรมและเทคโนโลยี มหาวิทยาลัยเทคโนโลยีราชมงคลอีสาน วิทยาเขตสกลนคร

Online Banking ทั่วโลก จากการจู่โจมด้วยเทคนิค SSL Stripping Attack แม้ผู้ให้บริการจะออกมาตรการรักษาความปลอดภัยเพิ่มขึ้น ด้วยการใช้ระบบยืนยันตัวตนแบบสองปัจจัย (Two-factor Authentications) โดยอาศัย One Time Password (OTP) ก็ตาม แต่แฮ็กเกอร์ก็สามารถพัฒนาเทคนิคอื่น อย่างการใช้ Zeus Trojan Horse หรือ Zbot [1] ร่วมกับ SSL Stripping Attack ในการจู่โจมระบบ SMS OTP จนประสบความสำเร็จ โดยมีกรณีการโจมตีที่เกิดขึ้นจริงกับธนาคารพาณิชย์หลายแห่งในประเทศไทยมาแล้ว [2] ถือเป็นวิกฤตที่ต้องแก้ไขอย่างเร่งด่วน

ปัจจุบันมีวิธีการมากมายถูกนำเสนอมาเพื่อแก้ไขปัญหา แต่วิธีการเหล่านั้นไม่ได้ทำการประเมินอย่างดี ต่างถูกประเมินโดยผู้เสนอเท่านั้น และแนวทางของการแก้ไขปัญหามีหลากหลาย จึงเกิดคำถามว่าวิธีการแก้ไขปัญหาเหล่านั้นมีประสิทธิภาพในการป้องกันการโจมตีจริงหรือไม่ ซึ่งจากงานวิจัยที่เสนอการประเมินปัญหาการโจมตีด้วยการเปลือยเอสเอสแอล [3] ค้นพบว่าการโจมตีสามารถกระทำได้ง่ายโดยไม่มีข้อจำกัดทางแพลตฟอร์ม แฮ็กเกอร์ที่มีความเชี่ยวชาญ (Professional Hacker) สามารถปรับปรุงโค้ดเพื่อต่อยอดความสามารถได้ ส่งผลให้วิธีการป้องกันที่เสนอก่อนหน้านี้อาจจะไม่ได้ผล เพราะป้องกันการโจมตีได้เพียงแฮ็กเกอร์มือสมัครเล่น (Script Kiddies) เท่านั้น โดยได้เสนอแนวคิดในการจู่โจมวิธีป้องกันที่ถูกนำเสนอมาแล้ว แต่ยังไม่ได้ทำการประเมินวิธีการป้องกันจริง

งานวิจัยนี้จึงเสนอขึ้นเพื่อประเมินวิธีการป้องกันการโจมตีจาก SSL Stripping Attack ที่ได้นำเสนอมาแล้ว ด้วยการวิเคราะห์และทดลองบน Test-bed โดยการอาศัยการแก้ไข Python Script ของ SSL Strip ในระดับ Professional Hacker จู่โจมวิธีการป้องกันเพื่อพิสูจน์ว่าจะเกิดปัญหาจากการโจมตีหรือไม่ ซึ่งผลที่ได้จะชี้ให้เห็นถึงขีดความสามารถของการโจมตีและข้อจำกัดของวิธีป้องกันที่ถูกนำเสนอ เพื่อนำไปใช้เป็นแนวทางในการออกแบบและปรับปรุงวิธีการป้องกันการโจมตีรูปแบบใหม่ที่มีประสิทธิภาพต่อไป

2. ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 การรักษาความมั่นคง Web Application ด้วย SSL

โปรแกรมประยุกต์เว็บที่มีความมั่นคงปลอดภัยนั้น ระบบจำเป็นจะต้องมีการสื่อสารข้อมูลอยู่บนโพรโทคอล HTTPS โดยอาศัย Secure Socket Layer (SSL) หรือเรียกอีกชื่อว่า Transport Layer Security (TLS) [4] เพื่อใช้เข้ารหัสการรับส่งข้อมูลรวมถึงการพิสูจน์ตัวตนในการสื่อสารระหว่างเซิร์ฟเวอร์และไคลเอนท์

SSL เป็นวิธีที่สามารถป้องกันการโจมตีได้ผลมาจนถึงทุกวันนี้ แต่การที่ SSL ไม่สามารถป้องกันการโจมตีได้ในบางกรณีเนื่องมาจาก SSL ไม่แข็งแกร่งพอในทางเทคนิค ซึ่งการโจมตีที่ผ่านมาไม่ว่าจะเป็น SSL Sniff [5] หรือ SSL Strip จะอาศัยการที่ผู้ใช้ไม่ให้ความร่วมมือ กล่าวคือผู้ใช้ผิดพลาดที่จะเข้าใจเทคโนโลยีที่ใช้ในการป้องกันอย่างแท้จริง จึงเกิดประเด็นคำถามขึ้นว่า ในเมื่อแฮ็กเกอร์โจมตีระบบโดยใช้ SSL Strip หลอกลวงผู้ใช้งานจนกระทั่ง SSL ซึ่งเป็นเทคนิครับมือที่ดีที่สุดถูกทำลายลงเนื่องจากความผิดพลาดของผู้ใช้ เมื่อมีนักวิจัยท่านอื่นพยายามที่จะแก้ไขปัญหาต่อโดยการคิดวิธีป้องกันขึ้นมาใหม่ คำถามที่ตามมาก็คือ แล้วผู้ใช้จะให้ความร่วมมือหรือไม่

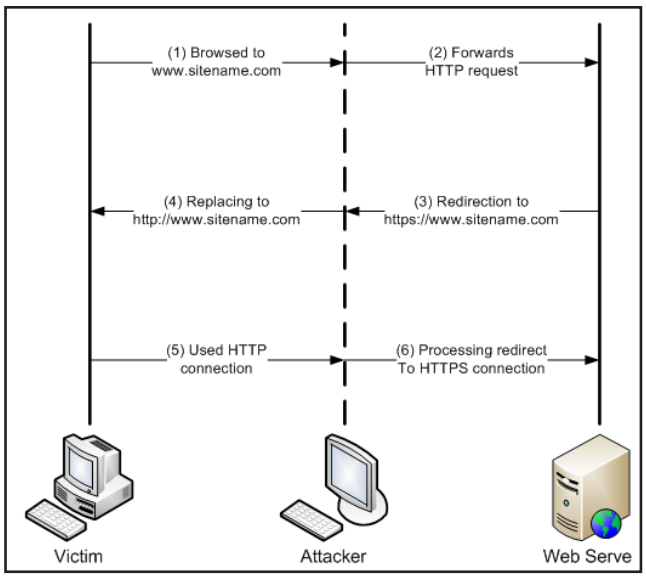
2.2 การจู่โจม SSL

การใช้งาน SSL สามารถถูกโจมตีได้จากเทคนิคการแทรกกลางการสื่อสาร (Man in the Middle: MITM) โดยอาศัยเครื่องมืออย่าง SSL Sniff และ SSL Strip เข้าร่วม ซึ่งพบว่าปัญหาของ SSL Sniff เกิดจากการไม่ตระหนักถึง session ของ SSL ที่ไม่สมบูรณ์ของผู้ใช้ ส่วนปัญหาของ SSL Strip นั้น เกิดจากผู้ใช้ไม่สนใจว่า session ของ SSL ทำงานอยู่หรือไม่

จากการศึกษาพบว่า การโจมตีด้วยเทคนิคการเปลือยเอสเอสแอล (SSL Stripping Attack) นี้ ถูกใช้งานอย่างแพร่หลายในการโจมตี Online Banking เนื่องจากจุดอ่อนที่เว็บเบราว์เซอร์ไม่สามารถตรวจสอบหรือแจ้งเตือนความผิดพลาดได้

SSL Stripping Attack ถูกสาธิตใน Black Hat Conference ปี ค.ศ. 2009 โดย Marlinspike [6] โปรแกรมที่ใช้จู่โจมคือ SSL Strip ถูกพัฒนาบนภาษา Python ซึ่งอาศัยวิธีโจมตีแบบแทรกกลางการสื่อสารร่วมกับการโจมตีแบบ SSL Stripping Attack ควบคู่กัน โดยการโจมตีเว็บไซต์ซึ่งทำงานบน HTTPS ที่ถูกเข้ารหัสในระหว่างการสื่อสารนั้น เมื่อเหยื่อถูกโจมตี เว็บเบราว์เซอร์จะถูกบังคับให้ใช้โพรโทคอล HTTP ในการสื่อสารแทน โดยข้อมูลต่างๆ ที่ถูกส่งไปที่เว็บเซิร์ฟเวอร์จะถูกส่งผ่านไปยังเครื่องของผู้โจมตีก่อน ซึ่งผู้โจมตีสามารถดักจับข้อมูลของเหยื่อได้อย่างง่ายดาย เนื่องจากข้อมูลที่

สื่อสารบน HTTP จะอยู่ในรูปของ Clear Text ที่สามารถเข้าใจได้ จากนั้นการทำงานในขั้นตอนต่อไป SSL Stripping Attack ก็จะนำข้อมูลของเหยื่อมาเข้ารหัสด้วยโพรโทคอล HTTPS แล้วส่งต่อไปยังเว็บเซิร์ฟเวอร์ ด้วยเหตุนี้จึงทำให้เว็บเซิร์ฟเวอร์ไม่สามารถตรวจสอบได้ว่าการสื่อสารที่เกิดขึ้นจริงในขณะนั้นกระทำอยู่บนโพรโทคอล HTTP หรือ HTTPS รวมถึงผลของการโจมตีที่เว็บเบราว์เซอร์ของเครื่องเหยื่อก็จะไม่สามารถตรวจสอบหรือแสดงข้อความแจ้งเตือนความผิดพลาดได้ เนื่องจากเครื่องของเหยื่อสามารถสื่อสารกับเว็บเซิร์ฟเวอร์ได้ตามปกติ เพียงแต่เป็นการสื่อสารที่ถูกบังคับให้อยู่บนโพรโทคอล HTTP แทนที่จะเป็นโพรโทคอล HTTPS ซึ่งมีความปลอดภัย ดังแสดงในภาพที่ 1



ภาพที่ 1 *รูปแบบการโจมตีของ SSL Stripping Attack*

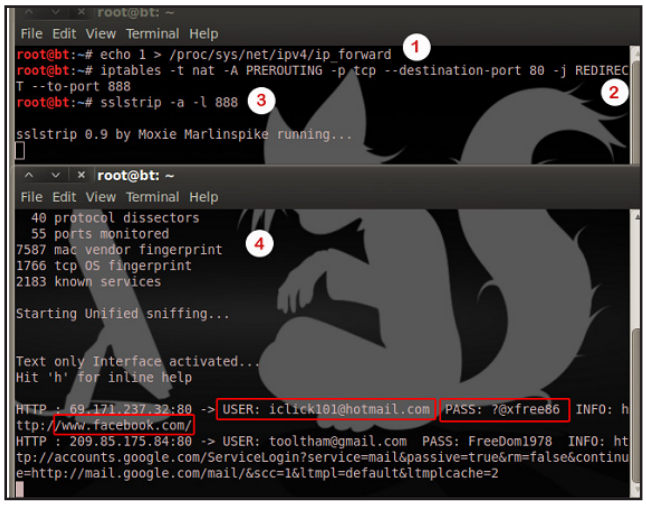
โดยขั้นตอนของการโจมตี SSL มีดังต่อไปนี้

1) โจมตีเพื่อแทรกกลางการสื่อสารระหว่างเหยื่อกับเว็บเซิร์ฟเวอร์ แล้วกำหนดให้ระบบทำการส่งต่อข้อมูลของเหยื่อที่เข้ามาไปยังเครื่องผู้โจมตีก่อนส่งผ่านไปที่เว็บเซิร์ฟเวอร์

2) กำหนดให้ข้อมูลที่เข้ามายังเครื่องผู้โจมตีทาง Port หมายเลข 80 ให้ส่งต่อไปที่ Port หมายเลขอื่น

3) ใช้คำสั่ง sslstrip เพื่อโจมตีเป้าหมาย ซึ่งตามปกติบนเว็บเบราว์เซอร์จะแสดงโพรโทคอล HTTPS แต่เมื่อถูกโจมตีแล้วจะส่งผลให้บนเว็บเบราว์เซอร์ถูกบังคับให้ใช้ HTTP แทน

4) ใช้ ettercap เพื่อแสดงข้อมูลของเครื่องเหยื่อที่สามารถดักจับได้ เช่น ชื่อบัญชีผู้ใช้และรหัสผ่านที่ใช้ในการตรวจสอบสิทธิ์ในการเข้าใช้งานระบบ



ภาพที่ 2 *คำสั่งที่ใช้ในการโจมตีของ SSL Stripping Attack*

2.3 งานวิจัยที่เกี่ยวข้องกับการแก้ไขปัญหา SSL Strip

อภิรักษ์ และ สมนึก [3] ได้ทำการประเมินปัญหาของการจู่โจมด้วยการเปลือยเอสเอสแอล โดยทดสอบโจมตีกลุ่มตัวอย่างเว็บไซต์ที่ประกอบด้วย Online Banking, Web Mail และ Social Net-work บนแพลตฟอร์มที่แตกต่างกันทั้งฮาร์ดแวร์ เว็บเบราว์เซอร์ และระบบปฏิบัติการ พบว่าแม้โปรแกรมประยุกต์เว็บจะมีการรักษาความปลอดภัยด้วย SSL หรือใช้ SSL ร่วมกับการป้องกันรูปแบบอื่น อย่างเช่นการเข้ารหัสข้อมูล (Message Encryption) หรือมีแฟลตฟอร์มที่แตกต่างกัน ระบบการให้บริการเหล่านั้นก็ยังสามารถถูกโจมตีด้วย SSL Stripping Attack ได้ ซึ่งการศึกษานี้เป็นเพียงการประเมินปัญหาการโจมตี SSL ในเชิงลึกและนำเสนอเพียงแนวคิดในการโจมตีวิธีป้องกันปัญหาเท่านั้น แต่ยังไม่ได้ทำการทดสอบเพื่อประเมินวิธีการป้องกันการโจมตีจริง

ปัจจุบันมีวิธีการแก้ไขปัญหาการโจมตีถูกเสนอขึ้นมากมาย แต่ส่วนใหญ่มักพบว่าวิธีการดังกล่าวเหล่านั้นยังขาดประสิทธิภาพและมีข้อบกพร่องที่ควรแก้ไขอยู่ ดังเช่นระบบ HProxy [7] ทำได้เพียงตรวจสอบการโจมตีและพบว่าบางกรณีระบบไม่ทำการแจ้งเตือนในขณะที่เว็บไซต์กำลังถูกโจมตี (False Negative) ระบบ SSLock [8] มีข้อเสียเรื่องมาตรฐานในการพัฒนาระบบ ซึ่งการปรับใช้กับเว็บไซต์เดิมมีความยุ่งยาก และป้องกันได้เพียง Domain Name ที่ถูกกำหนดเอาไว้เท่านั้น ระบบ HTTPSLock [9] หากระบบกำลังแจ้งเตือนการโจมตีจะไม่สามารถใช้งานเว็บไซต์ได้ตามปกติ เนื่องจากหน้าเพจเดิมถูกเปลี่ยนเป็นหน้าเพจเพื่อแสดงการแจ้งเตือนการโจมตีแทน และระบบ SSLight [10] สามารถ

ทำได้เพียงการแจ้งเตือนการโจมตี รวมถึงไม่รองรับการทำงานทุกเว็บเบราว์เซอร์

ด้วยข้อจำกัดของวิธีการป้องกันดังกล่าวข้างต้น จึงมีเพียงวิธีการแก้ไขปัญหาดังต่อไปนี้ที่ยังเป็นวิธีการที่น่าสนใจและควรถูกทำการประเมิน

Hodges และคณะ [11] ได้เสนอ HTTP Strict Transport Security (HSTS) ถูกพัฒนาจาก Force HTTPS ปัจจุบันถูก IETF กำหนดให้เป็น RFC 6797 เพื่อแก้ไขปัญหาการโจมตีโพรโทคอล HTTPS อย่างเช่น SSL Stripping Attack โดยอาศัยการกำหนดการตอบกลับของ HTTP Header ในการทำงานนั้น เมื่อเว็บเบราว์เซอร์ตรวจพบ HTTP Header ชื่อ Strict-Transport-Security ที่เป็นค่าเวลาในการกำหนดการบังคับใช้ HTTPS เว็บเบราว์เซอร์ก็จะบังคับใช้ HTTPS ตามเวลาที่ได้กำหนดไว้ ปัจจุบันมีเพียง Google Chrome และ Mozilla Firefox เท่านั้นที่รองรับการทำงาน และเบื้องต้นถูกจำกัดการป้องกันเฉพาะเว็บไซต์ที่อยู่ใน HSTS Preload List จำนวน 22 เว็บไซต์ เช่น google.com, lastpass.com, neg9.org, paypal.com และ stripe.com เท่านั้น แม้ว่า HSTS จะเปิดให้สามารถกำหนดรายชื่อเว็บไซต์อื่นที่ไม่ปรากฏใน Preload List ได้ แต่ยังพบปัญหาความยุ่งยากและความไม่เป็นมิตรต่อผู้ใช้งานเมื่อต้องกำหนดค่าใหม่

ณัฐวุฒิ และสมนึก [12] เสนอการแก้ไขปัญหาการโจมตี HTTPS ด้วยการเก็บรายชื่อเว็บไซต์ที่ต้องการป้องกันไว้ใน Bookmark ของเว็บเบราว์เซอร์ เพื่อความสะดวกในใช้งานและรองรับการทำงานในทุกเว็บเบราว์เซอร์ แต่พบว่าจะเกิดความยุ่งยากในการบันทึก URL ลงใน Bookmark ใหม่ทุกครั้ง เมื่อมีการใช้งานเว็บไซต์ที่ไม่เคยถูกบันทึก และกรณีที่ใช้งานหลายเว็บเบราว์เซอร์ก็จะสร้างความยุ่งยากเพิ่มขึ้นเมื่อต้องทำการเพิ่มข้อมูลใหม่ลงในทุกโปรแกรม

Puangpronpitag และ Sriwiboon [13] ได้เสนอ ISAN-HTTPS Enforcer ซึ่งถูกพัฒนาบนภาษา JavaScript ให้เป็น Application Programming Interface (API) ที่สามารถทำงานได้บนเว็บเบราว์เซอร์ที่รองรับ JavaScript โดยผู้ใช้ไม่ต้องปรับเปลี่ยนหรือติดตั้งโปรแกรมส่วนขยาย (Plug-in) เพิ่มทำงานได้บนแพลตฟอร์มที่มีความหลากหลาย ทั้งฮาร์ดแวร์เว็บเบราว์เซอร์และระบบปฏิบัติการ แต่มี Over-head ของเวลาการตอบสนอง (Response Time) ซึ่งเกิดจากการประมวลผลของ API ที่บังคับให้เว็บเบราว์เซอร์ติดต่อกลับไปยังเซิร์ฟเวอร์ด้วยโพรโทคอล HTTPS อีกครั้ง และแฮ็กเกอร์สามารถแก้ไข Python Script ของเทคนิค SSL Stripping Attack เพื่อทำการปลด JavaScript Tag ที่ใช้ในการป้องกันด้วยระบบนี้ออกได้

ACIS Professional Center [14] พัฒนา SSL StripGuard บน Android และ IOS เพื่อตรวจสอบการโจมตีจาก SSL Stripping Attack ในเครือข่ายไร้สาย โดยทำการติดต่อไปยัง SSL Server ที่กำหนดไว้เพื่อตรวจสอบการสื่อสารว่าอยู่บน HTTPS หรือไม่ ซึ่งเป็นแค่การตรวจสอบการโจมตี และต้องตรวจสอบก่อนใช้งานเครือข่ายทุกครั้ง เนื่องจากไม่ใช่การทำงานแบบ Real Time รวมถึงการใช้ SSL Server เพียงเครื่องเดียวโดยไม่มีการสุ่มเลือกหรือเปลี่ยนแปลง SSL Server ที่ใช้ตรวจสอบนี้ เพียงแฮ็กเกอร์ปรับแก้ Python Script ของ SSL Strip ก็สามารถ Bypass การป้องกันได้

อภิรักษ์ และสมนึก [15] ได้เสนอ Click2Enforce ซึ่งเป็นโปรแกรมส่วนขยาย (Browser Extension) ทำงานบน Google Chrome ที่สามารถทำการป้องกันการโจมตีจาก SSL Stripping Attack ได้ในรูปแบบอัตโนมัติจากการตั้งค่าบัญชีรายชื่อเว็บไซต์ใน White List และแบบผู้ใช้เป็นผู้กำหนดเองอย่างมีประสิทธิภาพ และมีความเป็นมิตรกับผู้ใช้งาน แต่วิธีป้องกันดังกล่าวยังเป็นเพียง Prototype ต้นแบบที่ถูกพัฒนาขึ้นสำหรับ Google Chrome เท่านั้น

3. การประเมินวิธีการป้องกัน SSL ที่ถูกนำเสนอ

ในการวิจัยครั้งนี้ เทคนิควิธีในการประเมินระบบที่ถูกนำมาทดสอบนั้น จะเลือกใช้เทคนิคของการ Measurement มาเป็นวิธีในการประเมินประสิทธิภาพของระบบ โดยกระทำในลักษณะการทำ Test-bed Network บนระบบที่กำหนดเอาไว้ภายใต้สภาพแวดล้อมที่เป็นสภาพแวดล้อมของเงื่อนไขจริง ซึ่งการทดสอบในลักษณะนี้มีความต้องการวัสดุ อุปกรณ์ โค้ด เงื่อนไขและเวลาที่ใช้ในการทำการทดลองจริง โดยมีการเฝ้า Monitor เพื่อสังเกตผลของการทดลองเป็น Key Point สำคัญ ซึ่งจะได้ผลลัพธ์ที่เหมือนจริงจากการประเมินประสิทธิภาพของระบบที่ถูกนำมาทำการทดลอง

3.1 เกณฑ์ในการประเมิน

เกณฑ์การประเมิน (Evaluation Criteria) ที่ถูกใช้เพื่อพิจารณาวิธีการป้องกันการโจมตีประกอบด้วย

1) ประสิทธิผลในการป้องกัน (Effectiveness) วิธีการแก้ไข

ปัญหาที่ถูกนำเสนอนั้นสามารถที่จะป้องกัน (Prevent) หรือทำได้เพียงแค่ตรวจสอบ (Detect) การโจมตีเท่านั้น โดยการทดสอบโจมตีวิธีที่ได้นำเสนอซึ่งถูกเลือกมาเป็นกลุ่มตัวอย่างด้วย SSL Stripping attack แล้วสังเกตประสิทธิภาพในการป้องกัน เพราะวิธีการที่ถูกนำเสนอเพื่อใช้ในการแก้ไขปัญหาที่แท้จริง จำเป็นจะต้องสามารถกระทำได้ทั้งการตรวจสอบและการป้องกันการโจมตี

2) ความเป็นมิตรต่อผู้ใช้ (User Friendliness) โดยทำการทดสอบระบบป้องกันการโจมตีว่ามีความยุ่งยากหรือซับซ้อนในการใช้งานหรือไม่ ซึ่งพิจารณาจากค่าเวลาในการทำงาน (Execution Time) ที่ถูกใช้เพื่อเข้าถึงการใช้งานวิธีป้องกันการโจมตี โดยอาศัย GOMS Model พื้นฐานแต่มีความน่าเชื่อถืออย่าง Keystroke-level Model (KLM) มาเป็นเครื่องมือทดสอบ ซึ่งพบว่าหากมีความยุ่งยากในการใช้งานแล้ว มักจะส่งผลให้ผู้ใช้ไม่ยอมรับหรือให้ความร่วมมือกับวิธีการแก้ไขปัญหาเหล่านั้น ถึงแม้จะเป็นวิธีป้องกันที่มีประสิทธิภาพก็ตาม

3) ข้อจำกัดในการป้องกันเพียงแค่แฮ็กเกอร์มือสมัครเล่น (Script-kiddy-only Limitation) การป้องกันสามารถรับมือกับผู้โจมตีในระดับ Professional Hacker ได้หรือไม่ หรือทำได้เพียงแค่ป้องกันการใช้เครื่องมือโจมตีของ Script Kiddies เท่านั้น โดยการดัดแปลงแก้ไขโค้ดการโจมตีของวิธี SSL Stripping Attack บนภาษา Python เพียงเล็กน้อย เนื่องจากแฮ็กเกอร์ที่มีเป้าหมายในการมุ่งโจมตีระบบให้บริการมักมีความรู้ความเชี่ยวชาญสูง จึงจำเป็นที่วิธีการป้องกันจะต้องปกป้องระบบได้มากกว่าการใช้เครื่องมือการโจมตีขั้นพื้นฐาน

4) ความครอบคลุมในแพลตฟอร์ม (Platform Coverage) ระบบรองรับทำงานบนแพลตฟอร์มที่หลากหลายได้หรือไม่ โดยทดสอบว่าวิธีการป้องกันที่ถูกนำเสนอมานั้น สามารถทำงานได้บนแพลตฟอร์มใดบ้าง เพราะปัญหาการถูกโจมตีด้วย SSL Stripping Attack ไม่ได้ถูกจำกัดเพียงแค่แพลตฟอร์มของระบบปฏิบัติการ เว็บเบราว์เซอร์ หรือฮาร์ดแวร์แพลตฟอร์มใดแพลตฟอร์มหนึ่งเท่านั้น

3.2 วิธีการป้องกัน SSL ที่ถูกนำมาทดสอบ

ในงานวิจัยนี้ได้เลือกระบบการป้องกันมาทำการทดสอบโดยมีการพิจารณาคัดเลือกดังนี้

1) HSTS ถูกเลือกมาทำการทดสอบเนื่องจากปัจจุบันเป็นหนึ่งในมาตรฐานของ IETF (RFC 6797) ที่ถูกเสนอให้ใช้ป้องกัน SSL Stripping Attack และปัจจุบันถูกใช้งานจริงบน Google Chrome และ Mozilla Firefox ภายใต้การสนับสนุนของ Google และ PayPal

2) ISAN-HTTPS Enforcer ป้องกันการโจมตีจาก SSL Stripping Attack ได้บนทุกแพลตฟอร์มของฮาร์ดแวร์ ระบบปฏิบัติการ และเว็บเบราว์เซอร์ มีความเป็นมิตรต่อผู้ใช้โดยไม่ต้องติดตั้งโปรแกรมส่วนขยายเพิ่ม

3) SSL StripGuard เนื่องจากพบว่าเป็น Application เดียวที่ใช้ในการปกป้อง Mobile Device และมีปริมาณยอด Download สูงที่สุดในประเทศทั้งบน App Store ของ IOS และ Google Play ของ Android

3.3 เครื่องมือที่ใช้ในการทดสอบ

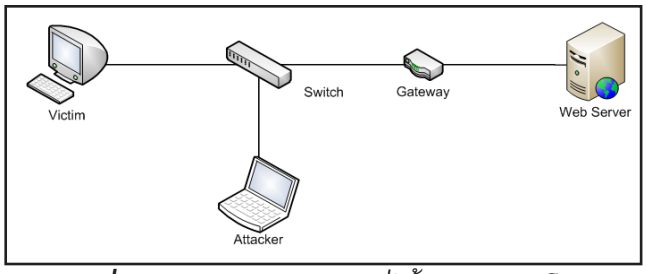
1) โปรแกรมที่ใช้ในการโจมตี จะใช้ SSL Strip ในการจู่โจมระบบ โดย Ettercap และ Wireshark จะถูกใช้ในการดักจับข้อมูลจากการโจมตี ซึ่งเครื่องมือดังกล่าวถูกติดตั้งบน Back Track 5

2) เว็บเบราว์เซอร์ ประกอบด้วย Google Chrome ใช้ทดสอบ HSTS และ ISAN-HTTPS Enforcer ส่วน Safari และ Android HTML Webkit ใช้เพื่อทดสอบ SSL StripGuard

3) ระบบปฏิบัติการที่ใช้ทดสอบ ประกอบด้วย Microsoft Windows 7 (PC Desktop), Back Track 5 R3 (Notebook), Android 4.0.4 (Sony Arc S) และ IOS 6.1.2 (iPhone 4S)

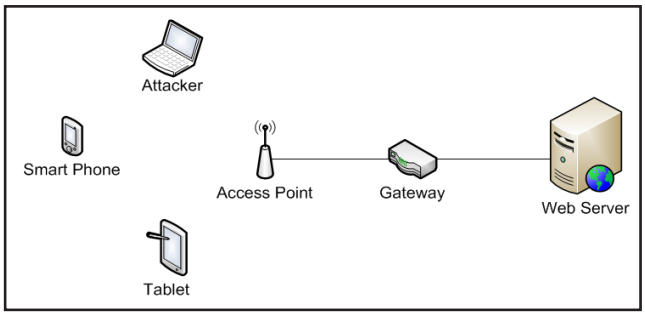
3.4 สภาพแวดล้อมที่ใช้ในการทดลอง

สภาพแวดล้อมที่เป็น Test-bed ในการทดสอบโจมตีแบ่งออกเป็น Wire Network ใช้สำหรับทดสอบการโจมตีบน PC Desktop ดังภาพที่ 3 และ Wireless Network ใช้สำหรับทดสอบการโจมตีบน Mobile Device ดังภาพที่ 4



ภาพที่ 3 *ระบบ Wire Network ที่ใช้ทดสอบการโจมตี*

คอมพิวเตอร์ที่ใช้ในการโจมตี คือ Intel ® Core ™ 2 Duo 2.66 GHz RAM 4 GB ระบบปฏิบัติการ Linux Back Track 5 R3 สำหรับเครื่องที่ถูกโจมตีประกอบด้วย PC Desktop Intel ® Core ™ i5 3.30 GHz Ram 4 GB ระบบปฏิบัติการ



ภาพที่ 4 *ระบบ Wireless Network ที่ใช้ทดสอบการโจมตี*

Windows 7 Professional, Apple iPhone 4S 16GB ระบบปฏิบัติการ IOS 6.1.2 และ Sony Xperia Arc S ระบบปฏิบัติการ Android 4.0.4 Ice-cream Sandwich

3.5 แนวคิดในการทดสอบ HSTS

ระบบ HSTS นั้น แม้จะมีประสิทธิภาพในการป้องกันการโจมตีแต่ยังพบข้อจำกัด คือ 1) รองรับการทำงานเพียง Google Chrome และ Mozilla Firefox 2) ยุ่งยากและไม่เป็นมิตรต่อผู้ใช้ เมื่อต้องกำหนดเว็บไซต์ที่ต้องการป้องกันด้วยตนเอง กล่าวคือ HSTS ประสบความสำเร็จในการป้องกันแต่กลับล้มเหลวในด้านของความเป็นมิตรกับผู้ใช้

ในการประเมินประสิทธิภาพจึงใช้วิธี Keystroke-level Model (KLM) [16] มาทดสอบ เพื่อชี้ให้เห็นข้อบกพร่องในแง่มุมของ User Friendliness เทียบกับวิธีการใช้ “https://” และ Bookmark ทดสอบบน Google Chrome โดยการกำหนดเว็บไซต์ที่ไม่มีอยู่ใน Preload List ของ HSTS งานวิจัยนี้ได้เลือก www.facebook.com ซึ่งเป็นเว็บไซต์ Social Network ที่ได้รับความนิยม เป็นตัวอย่างในการทดลอง โดยการทดสอบจะเริ่มตั้งแต่การป้อนข้อมูลใน Google Chrome เพื่อทำการกำหนดค่าการป้องกัน และสิ้นสุดเมื่อเรียกใช้งานเว็บไซต์แล้วระบบดังกล่าวสามารถป้องกันเว็บไซต์ที่กำหนดได้

3.6 แนวคิดในการทดสอบ ISAN-HTTPS Enforcer

จากความสามารถ Python Script ของ SSL Strip ซึ่งปลด Tag ที่บังคับใช้โพรโทคอล HTTPS ในการสื่อสารออกได้นั้น การทดสอบนี้จึงอาศัยหลักการดังกล่าวเพื่อทำลายการป้องกันของ ISAN-HTTPS Enforcer ซึ่งพบว่าระบบป้องกันนี้ถูกพัฒนาด้วย JavaScript เป็น API เก็บไว้ที่เว็บเซิร์ฟเวอร์ แต่จะถูกเรียกใช้และประมวลผลทุกครั้งที่ไคลเอนต์ เพื่อตรวจสอบและบังคับการใช้ HTTPS ดังนั้นการเรียกใช้งาน API ด้วย Tag บน HTML จึงเป็นช่องโหว่ที่ใช้โจมตีได้

```
SET RemoveTag = null
SET data = null

READ(all tag in HTML, data)
IF data != null THEN
  SET PositionOfJavaScriptTag
  IF PositionOfJavaScriptTag != null THEN
    RemoveTag = RE.COMPILE(PositionOfJavaScriptTag)
    IF RemoveTage != null THEN
      data = RemoveTag.SUB(null, data)
    END
  END
END

RETURN data
```

ภาพที่ 5 *อัลกอริธึมที่ใช้ในการโจมตี HTTPS Enforcer*

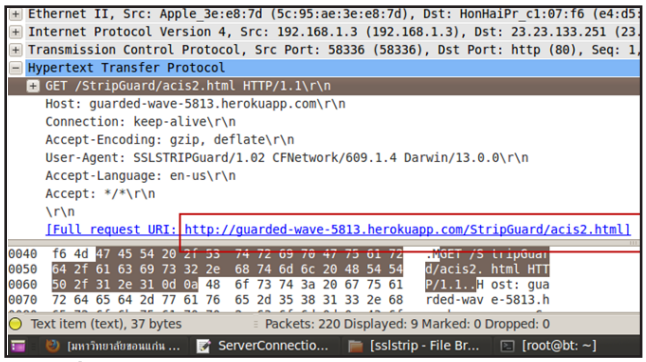
```
removeTag = re.compile(r‘<script.*?isan-httpsenforcer
    .js”></script>’, re.IGNORECASE)
    data = removeTag.sub(‘’, data)
return data
```

ภาพที่ 6 *โค้ดที่ใช้เพื่อปลดระบบการป้องกัน*

ในการทดสอบจะนำ Regular Expression Operations (re) [17] ของภาษา Python มาใช้ ซึ่งมีแนวคิดการโจมตีดังอัลกอริธึมในภาพที่ 5 แล้วทำการเขียนโค้ดภาษา Python เพิ่มเข้าไปในไฟล์ที่ใช้ในการติดต่อไปยังเว็บเซิร์ฟเวอร์ของ SSL Strip ใน Back Track R3 ดังภาพที่ 6

3.7 แนวคิดในทดสอบ SSL StripGuard

เมื่อใช้ Wireshark ดักจับ Packet ในระหว่างการสื่อสารที่เกิดขึ้นตั้งแต่ SSL StripGuard เริ่มทำงานจนกระทั่งการตรวจสอบการโจมตีเสร็จสิ้น พบว่าการทำงานจะเริ่มจากโหลด advertise.html ขึ้นมาแสดง ก่อนที่จะย้ายหน้าเพจ (Redirect) ไปยัง acis2.html เพื่อรอให้ผู้ใช้คลิกปุ่ม Check และเมื่อคลิกที่ปุ่ม แล้วโปรแกรมจะเรียกไปยังไฟล์ check2.html เพื่อตรวจสอบ ซึ่งหากถูกโจมตีด้วย SSL Stripping Attack หน้าเพจจะถูก Redirect ไปยังไฟล์ strip2.html เพื่อเตือนผู้ใช้ว่าไม่ปลอดภัย แต่ถ้าไม่ได้ถูกโจมตีก็จะทำการ Redirect ไปที่ secure2.html แทน เพื่อแจ้งว่าปลอดภัย โดยค้นพบว่าเงื่อนไขการตรวจสอบคือ การ request ไปยังเซิร์ฟเวอร์ที่กำหนดให้เป็น SSL Server (https://guarded-wave-5813.herokuapp.com/StripGuard) ดังภาพที่ 7 ถ้ามี Response กลับมาว่ามีการสื่อสารอยู่บน HTTPS จะแสดงว่าเครือข่ายมีความปลอดภัย แต่ถ้าหากอยู่บน HTTP ก็จะแจ้งเตือนว่าเครือข่ายไม่ปลอดภัย



ภาพที่ 7 *SSL StripGuard ติดต่อไปยัง SSL Server ที่กำหนด*

ในการทดสอบโจมตีจึงต้องปรับปรุงเงื่อนไขในการจู่โจมของ Python Script ใหม่ โดยปล่อยให้ SSL Strip ทำการปลด HTTPS ออกตามปกติ แต่ในขั้นตอนสุดท้ายของการโจมตีจะต้องกำหนดให้ตรวจสอบอีกครั้งว่ามี URL ที่ตรงกับ SSL Server ที่ถูกใช้เพื่อตรวจสอบหรือไม่ จากนั้นจึงทำการคืนค่าจาก HTTP มาเป็น HTTPS อีกครั้งเฉพาะ URL ของ SSL Server นี้ เพื่อหลบเลี่ยง (Bypass) ไม่ให้ SSL StripGuard ตรวจสอบพบว่ามีการโจมตีเกิดขึ้น ซึ่งจะต้องแก้ไขตามขั้นตอนนี้เท่านั้น เพื่อไม่ให้กระทบกับการโจมตีเว็บไซต์อื่น และป้องกันการเกิดวงรอบที่ไม่รู้จบ (Loop) ขึ้นใน Python Script ที่ใช้โจมตี ดังอัลกอริธึมที่แสดงในภาพที่ 8

```
SET BypassTag = null
SET data = null

READ(all tag in HTML, data)
IF data != null THEN
  SET URL_Of_SSLServer
  SET NewURL = “https://”+URL_Of_SSLServer
  IF URL_Of_SSLServer != null THEN
    BypassTag = RE.COMPILE(URL_Of_SSLServer)
    IF BypassTage != null THEN
      data = BypassTag.SUB(NewURL, data)
    END
  END
END

RETURN data
```

ภาพที่ 8 *อัลกอริธึมที่ใช้ในการโจมตี SSL StripGuard*

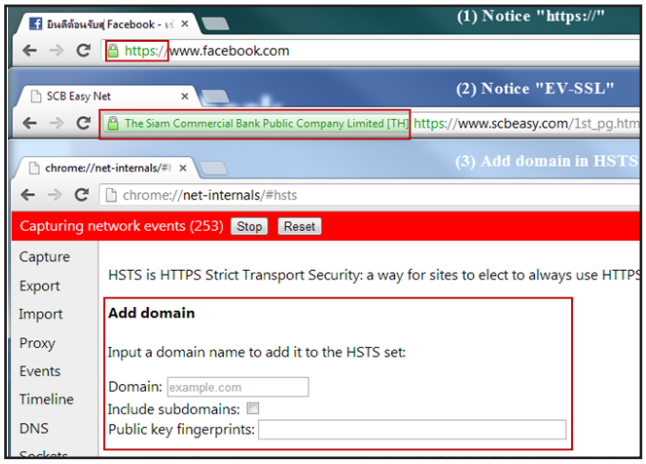
```
    bypass = re.compile(r‘http://guard’, re.IGNORECASE)
    data = bypass.sub(‘https://guard’, data)
return data
```

ภาพที่ 9 *โค้ดที่ใช้ในการจู่โจมเพื่อ Bypass ระบบ*

4. ผลการประเมินวิธีการป้องกัน SSL ที่ถูกนำเสนอ

4.1 ผลการประเมินระบบ HSTS

HSTS มีขั้นตอนที่ต้องปฏิบัติตาม ถ้าผู้ใช้ไม่ให้ความร่วมมืออาจเกิดปัญหาการจู่โจมขึ้นได้ถ้าเว็บไซต์นั้นไม่อยู่ใน Preload List ซึ่งพบว่าการเพิ่มเว็บไซต์อื่นเข้าไปในระบบก็อาจจะไม่ได้รับความร่วมมือจากผู้ใช้ เพราะมีความยุ่งยาก กล่าวคือถ้าผู้ใช้ไม่สังเกตแม้แต่ “https://” หรือ EV-SSL แล้วนั้น ผู้ใช้ก็อาจจะไม่ให้ความร่วมมือกับ HSTS เช่นกัน แต่ถ้าให้ความร่วมมือ ผู้ใช้เองก็อาจจะสังเกต “https://” หรือ EV-SSL อยู่แล้ว ดังนั้น HSTS จึงไม่มีความหมายหากผู้ใช้ตระหนักถึงปัญหาและใช้การสังเกต ดังงานวิจัยของ ณัฐวุฒิ ศรีวิบูลย์ และสมนึก พ่วงพรพิทักษ์ [13] จากภาพที่ 10 แสดงให้เห็นว่าการสังเกต “https://” หรือ EV-SSL ง่ายกว่าการใช้งาน HSTS



ภาพที่ 10 *การสังเกต “https://” หรือ EV-SSL และ HSTS*

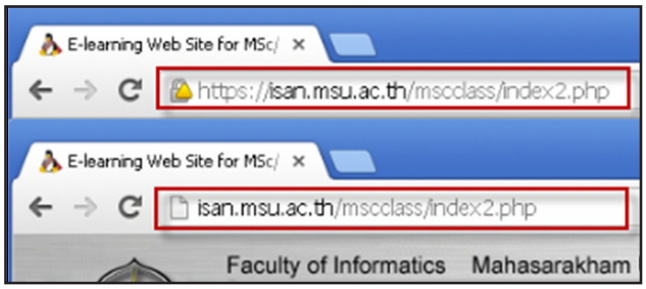
จากการทดสอบด้วย KLM พบว่าเวลาที่ใช้ใน HSTS นั้นมีค่าสูงกว่าวิธีอื่น แสดงให้เห็นว่าเมื่อต้องการความปลอดภัยในการใช้งานเว็บไซต์นั้น การพิมพ์ “https://” และใช้ Bookmark จะสะดวกและเข้าถึงการทำงานได้ง่ายกว่า HSTS โดยผลของการทดสอบจะแสดงเวลาทั้งหมดที่ถูกใช้ในการดำเนินกิจกรรมบน Google Chrome ดังตารางที่ 1

ตารางที่ 1 *ผลการทดสอบประสิทธิภาพด้วยวิธี KLM*

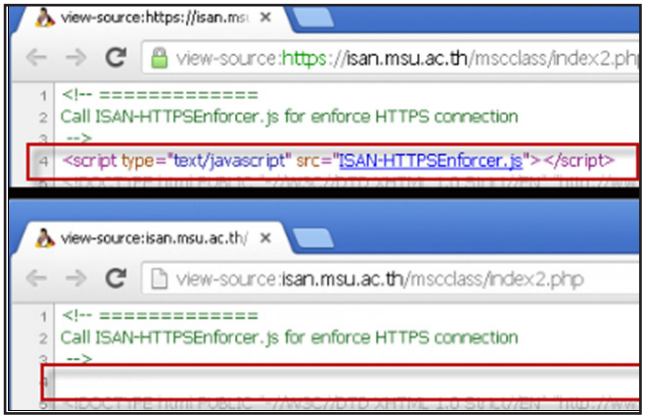
Solution	Total Time (Second)
Use “https://”	6.4
Bookmark	14.9
HSTS	19

4.2 ผลการประเมินระบบ ISAN-HTTPS Enforcer

การทดลองโจมตีเว็บไซต์ www.isan.msu.ac.th ซึ่งใช้ ISAN-HTTPS Enforcer ป้องกันพบว่า เมื่อทำการแก้ไข Python Script ที่ใช้โจมตีเพียง 2 บรรทัดก็สามารถปลดระบบการป้องกันออกได้อย่างสมบูรณ์ โดยไม่มีการแจ้งเตือนความผิดปกติให้ทราบ ดังผลการโจมตีในภาพที่ 11 และภาพที่ 12



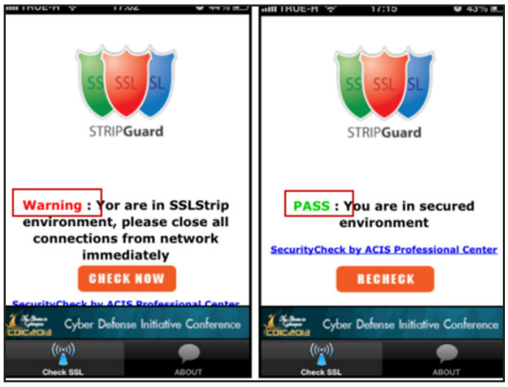
ภาพที่ 11 *ผลของการโจมตี ISAN-HTTPS Enforcer*



ภาพที่ 12 *ผลของการโจมตี ISAN-HTTPS Enforcer*

4.3 ผลการประเมินระบบ SSL StripGuard

การทดลอง Bypass ระบบตรวจสอบการโจมตี พบว่าเมื่อทำการแก้ไข Python Script แล้วจะสามารถ Bypass การโจมตีไม่ให้ SSL StripGuard ตรวจพบความผิดพลาดได้ ซึ่งผลการตรวจสอบของโปรแกรมจะแสดงว่ามีความปลอดภัยในการใช้งานอยู่เสมอ เมื่อคลิกตรวจสอบด้วยปุ่ม Check ถึงแม้เว็บไซต์ต่างๆ จะถูกโจมตีด้วยการเปลือยเอสเอสแอลอยู่ก็ตาม ดังภาพที่ 13



(a) ก่อน Bypass ระบบ (b) หลัง Bypass ระบบ

(c) เกิดการโจมตี SSL โดยที่ระบบไม่แจ้งเตือน

ภาพที่ 13 *ผลการโจมตีเพื่อ Bypass ระบบ SSL StripGuard*

5. วิจารณ์ผลการวิจัย

จากการทดสอบเพื่อประเมินวิธีการป้องกันการโจมตีพบว่า

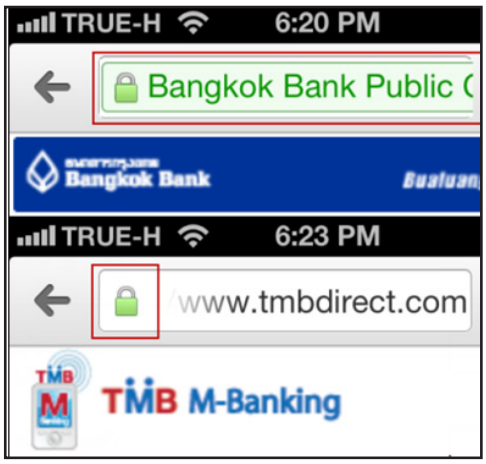
1) การออกแบบวิธีการป้องกันควรต้องคำถึงการป้องกันที่ไม่ได้สกัดกั้นเพียงผู้โจมตีระดับ Script Kiddies เท่านั้น เนื่องจาก Professional Hacker ที่มีประสบการณ์หากได้วิเคราะห์รูปแบบที่ใช้ในการป้องกันก็จะสามารถทำการปรับปรุงแก้ไขการโจมตีเพื่อทำลายวิธีป้องกันนั้นลงได้

2) ระบบ HSTS เมื่อถูก IETF กำหนดเป็น RFC แล้ว จะถูกยอมรับและนำไปใช้ในทุกเว็บเบราว์เซอร์ เป็นวิธีป้องกันที่ดี โดยเฉพาะกับเว็บไซต์ที่อยู่ใน Preload List แต่เป็นเรื่องยากที่จะใส่รายชื่อเว็บไซต์ทั้งหมดไว้ใน List นั้น และในด้าน User Friendliness ระบบนี้ไม่เหมาะกับผู้ใช้ที่ไม่ให้ความร่วมมือ แม้ผู้ใช้จะให้ความร่วมมือก็ไม่มีความจำเป็น เนื่องจากการใช้และสังเกต “https://” ทำได้ง่ายกว่า

3) ระบบ ISAN-HTTPS Enforcer นั้น แม้จะป้องกันการโจมตีได้อย่างมีประสิทธิภาพบนแพลตฟอร์มที่หลากหลาย และมีจุดเด่นใน User Friendliness แต่กลับมีช่องโหว่ที่สามารถทำลายระบบได้ด้วยการแก้ไขโค้ดโจมตีเพียงสองบรรทัด

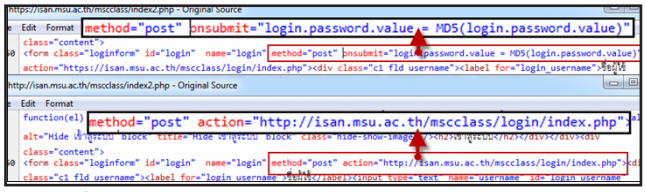
4) ระบบ SSL StripGuard เป็นทางเลือกเดียวบน Mobile Device ซึ่งมี User Friendliness ที่ไม่สูงนัก เนื่องจากต้องทำการติดตั้งและผู้ใช้ต้องให้ความร่วมมือ แต่ถ้าผู้ใช้ต้องติดตั้งและรันโปรแกรมแล้ว การสังเกต EV-SSL หรือ “https://” ดังภาพที่ 14 จะทำได้ง่ายกว่าเพราะสังเกตได้จากหน้าจอ

ของเว็บเบราว์เซอร์อยู่แล้ว จึงไม่มีความจำเป็นต้องใช้ รวมถึงขบวนการตรวจสอบมีรูปแบบที่เรียบง่ายเกินไปจนทำลายได้ด้วยการแก้ไขโค้ดการโจมตีเล็กน้อย

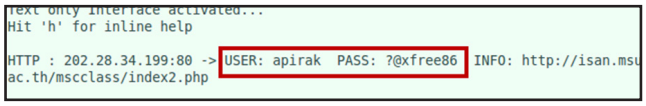


ภาพที่ 14 แสดงจุดสังเกต SSL บน Mobile Device

5) ในระบบที่ใช้ SSL ร่วมกับ Message Encryption นั้น พบว่า Python Script สามารถทำลายการป้องกันได้เช่นกัน ซึ่งการเข้ารหัสข้อมูลมักจะอาศัย Tag เพื่อเรียกใช้ External Script จากภายนอก โดยผลการทดสอบโจมตีระบบตามแนวคิดนี้แสดงไว้ดังภาพที่ 15 และภาพที่ 16



ภาพที่ 15 View Source ผลการโจมตี Message Encryption



ภาพที่ 16 ผลการดักจับข้อมูลหลังจากการโจมตี

ผลการทดลองเพื่อประเมินวิธีการแก้ไขปัญหาของการโจมตีด้วย SSL Stripping Attack แสดงไว้ดังตารางที่ 2

6. สรุปผลการวิจัย

SSL Stripping Attack เป็นภัยคุกคามร้ายแรงที่ส่งผลกระทบต่อโปรแกรมประยุกต์เว็บมาตั้งแต่อดีต จึงมีแนวทางการแก้ไขปัญหาถูกเสนอขึ้นมากมาย แต่วิธีเหล่านั้นกลับขาดการประเมินอย่างจริงจัง จึงไม่ปรากฏแน่ชัดว่าสามารถทำงานได้จริงหรือไม่ ในงานวิจัยนี้ได้เปิดเผยให้เห็นว่าวิธีแก้ไขปัญหาที่ได้รับการยอมรับอย่างระบบ HSTS แม้ประสบความสำเร็จในการป้องกัน แต่กลับล้มเหลวในด้านของความเป็นมิตรกับผู้ใช้ ซึ่งเห็นได้จากเวลาที่ถูกใช้ในการประเมินประสิทธิภาพด้วย KLM ที่ยังมีค่าสูงเมื่อเปรียบเทียบกับวิธีป้องกันรูปแบบอื่น ซึ่งส่งผลให้ผู้ใช้ในระดับทั่วไปไม่ยอมรับเพราะมีขั้นตอนการเข้าถึงที่ซับซ้อน, ระบบ ISAN-HTTP Enforcer ถึงแม้จะสามารถป้องกันการโจมตีได้บนทุกแพลตฟอร์ม และมีความเป็นมิตรกับผู้ใช้สูง แต่กลับถูกทำลายการป้องกันได้ด้วยการแก้ไขโค้ดที่ใช้ในการโจมตีเพียงสองบรรทัด ด้วยการอาศัยความสามารถพื้นฐานในการค้นหาและแทนที่ค่าของ Regular Expression บน Python และระบบ SSL StripGuard มีรูปแบบการทำงานที่เรียบง่ายเกินไป ซึ่งสามารถตรวจสอบอัลกอริธึมการทำงานได้ด้วยการดักจับ Packet ในการสื่อสารจนนำไปสู่การ Bypass การตรวจสอบได้ด้วยการแก้ไขโค้ดที่ใช้ในการโจมตีได้เช่นเดียวกัน ซึ่งผลของการทดลองแสดงให้เห็นว่าวิธีการป้องกันที่ถูกนำเหล่านี้ล้วนแต่ไม่สามารถบรรลุผลในด้านประสิทธิภาพของการป้องกันได้ดีไปกว่าการสอนให้

ตารางที่ 2 สรุปผลการประเมินวิธีการแก้ไขปัญหาการโจมตี

Solution	Experimental Results
HSTS	- ป้องกันการโจมตีเว็บไซต์ที่อยู่ใน List ถ้านอกเหนือจาก List นั้นต้อง add เพิ่ม - ไม่เป็นมิตรกับผู้ใช้ ยุ่งยากเกินไปที่ผู้ใช้ทั่วไปจะให้ความร่วมมือ - ป้องกัน Professional Hacker ได้ - รองรับเพียง Chrome และ Firefox บนแพลตฟอร์ม PC Desktop
ISAN-HTTPS Enforcer	- ป้องกันเว็บไซต์จากการโจมตี SSL - มีความเป็นมิตรกับผู้ใช้สูงเนื่องจากไม่ต้องทำอะไรเลย ระบบจัดการอัตโนมัติ - ป้องกันได้เพียง Script Kiddies เพราะ Professional Hacker แก้ไขโค้ดโจมตีได้ - รองรับทุกแพลตฟอร์ม
SSL StripGuard	- ทำได้เพียงตรวจสอบการโจมตี SSL - มีความเป็นมิตรกับผู้ใช้ที่ต่ำเนื่องจากต้องติดตั้งและให้ความร่วมมือในการตรวจสอบ - ป้องกันได้เพียง Script Kiddies เพราะ Professional Hacker แก้ไขโค้ดโจมตีได้ - ถูกพัฒนาให้รองรับเพียงแพลตฟอร์มที่เป็น Mobile Device

ผู้ใช้พิมพ์และเรียนรู้การสังเกต “https://” หรือ EV-SSL ทั้งสิ้น เนื่องจากการพิมพ์และสังเกตสามารถกระทำได้ง่าย และป้องกันการโจมตีได้แท้จริง แต่กลับพบปัญหาว่าผู้ใช้ส่วนใหญ่มักไม่ใช้วิธีดังกล่าวนี้เลย

ด้วยเหตุนี้แนวทางในการพัฒนาวิธีการป้องกันการโจมตีในอนาคต จึงควรที่จะทำได้ทั้งการตรวจสอบและป้องกันการโจมตี โดยไม่มีข้อจำกัดในแพลตฟอร์มของการทำงาน มีความเป็นมิตรกับผู้ใช้ อัลกอริธึมที่ใช้มีความซับซ้อนและมีประสิทธิภาพในการป้องกันการโจมตีจากแฮ็กเกอร์ที่มีความเชี่ยวชาญในระดับที่สามารถแก้ไขโค้ดการโจมตีได้

7. จรรยาบรรณในการวิจัย

ด้วยข้อจรรยาบรรณ การทดลองโจมตีด้วยเทคนิค SSL Stripping Attack นี้ไม่ได้มีวัตถุประสงค์ในการเจาะเข้าไปในระบบการให้บริการของเว็บไซต์ที่ใช้ในการทดสอบแต่เป็นการโจมตีเพื่อทำการทดสอบดักจับข้อมูลในระหว่างการสื่อสารเท่านั้น ซึ่งการทดลองไม่ได้ใช้ชื่อผู้ใช้หรือรหัสผ่านจริง และการทดลองกระทำอยู่บน Test-bed ที่ถูกควบคุมตามหลักจริยธรรม ของ Ethical Hacker อย่างเข้มงวด

8. เอกสารอ้างอิง

[1] Kaspersky Lab. “ZeuS on the Hunt.” Available online at http://www.securelist .com/en/analysis/204792107/ZeuS_on_the_ Hunt, 2012.

[2] IT24Hrs. “Fake Mobile SMS Banking Hack.” Available online at http://www.it24hrs.com/2013/fake-mobile-sms-banking-hack, 2013.

[3] A. Tooltham and S. Puangpronpitag. “The Evaluation of the SSL Stripping Attack Problem.” *In Proceedings of the National Conference on Computer Information Technologies*, Sakon Nakhon, Thailand, pp. 43-48, January 2013.

[4] T. Dierks and E. Rescorla. “The Transport Layer Security (TLS) Protocol Version 1.2.” IETF, RFC 5246, 2008.

[5] M. Marlinspike. “SSL Sniff.” Available online at http://www.thoughtcrime.org /software/sslsniff/, 2012.

[6] M. Marlinspike. “SSL Strip.” Available online at http://www.thoughtcrime.org /software/sslstrip/, 2012.

[7] N. Nikiforakis, Y. Younan and W. Joosen. “HProxy: client-side detection of SSL stripping attack.” *In Proceedings of the 7th international conference on Detection of intrusions, malware, and vulnerability assessment,* Bonn, Germany, pp. 200-218, July 2010.

[8] A. Fung and K. Cheung. “SSLock: sustaining the trust on entities brought by SSL.” *In Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security,* Beijing, China, pp. 204-213, April 2010.

[9] A. Fung and K. Cheung. “HTTPSLock: Enforcing HTTPS in Unmodified Browsers with Cached Javascript.” *In Proceedings of the 4th Network and System Security*, Melbourne, Australia, pp. 269-274, September 2010.

[10] D. Shin and R. Lopes. “An empirical study of visual security cues to prevent the SSL Stripping attack.” *In Proceedings of the 27th Annual Computer Security Applications Conference*, Orlando, Florida, pp. 287-296, December 2011.

[11] J. Hodges, C. Jackson and A. Barth. “HTTP Strict Transport Security (HSTS).” IETF, RFC 6797, November 2012.

[12] N. Sriwiboon and S. Puangpronpitag. “Detection & Protection Mechanisms Against SSL Strip Attacks.” *In Proceedings of 9th International Joint Conference on Computer Science and Software Engineering*, Bangkok, Thailand, pp. 25-30, 2012.

[13] S. Puangpronpitag and N. Sriwiboon. “Simple and Light-weight HTTPS Enforcement to Protect Against SSL Striping Attack.” *In Proceedings of 4th International Conference on Computational Intelligence, Communication Systems and Networks,* Phuket, Thailand, pp. 229-234, June 2012.

[14] ACIS Professional Center. “SSL StripGuard version 1.01.” April 2012.

[15] A. Tooltham and S. Puangpronpitag. "Click2Enforce: a Browser Extension to Protect against SSL Stripping Attacks." *Information Technology Journal*, Vol. 9 No. 2, July-December 2013.

[16] D. Kieras. "Using the Keystroke-Level Model to Estimate Execution Times." Available online at http://www.pitt.edu/~cmlewis/KSM.pdf, 1993.

[17] Python Software Foundation. "Regular expression opera-tions." Available online at http://docs.python.org/2/library/re.html, 2012.