กลไกสำหรับความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส

Mechanism for Securing WSDL Web Service

เจนจิรา หวังหลี

Janejira Wanglee

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญา
วิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์
มหาวิทยาลัยสงขลานครินทร์

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Computer Science
Prince of Songkla University
2557

ชื่อวิทยานิพนธ์ กลไกสำหรับความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส
ผู้เขียน นางสาวเจนจิรา หวังหลี
สาขาวิชา วิทยาการคอมพิวเตอร์

อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

.....
(ผู้ช่วยศาสตราจารย์ ดร.ลัดดา ปรีชาวีรกุล)

คณะกรรมการสอบ

.....ประธานกรรมการ
(ดร.นพมาศ ปักเข็ม)

.....กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.ลัดดา ปรีชาวีรกุล)

.....กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.ศิริรัตน์ วณิชโยบล)

.....กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.วิภาดา เวทย์ประสิทธิ์)

บัณฑิตวิทยาลัย มหาวิทยาลัยสงขลานครินทร์ อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษา ตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์

.....
(รองศาสตราจารย์ ดร.ธีระพล ศรีชนะ)
คณบดีบัณฑิตวิทยาลัย

ขอรับรองว่า ผลงานวิจัยนี้มาจากการศึกษาวิจัยของนักศึกษาเอง และได้แสดงความขอบคุณบุคคลที่มีส่วนช่วยเหลือแล้ว

ลงชื่อ……..
(ผู้ช่วยศาสตราจารย์ ดร.ลัดดา ปรีชาวีรกุล)
อาจารย์ที่ปรึกษาวิทยานิพนธ์

ลงชื่อ……
(นางสาวเจนจิรา หวังหลี)
นักศึกษา

ข้าพเจ้าขอรับรองว่า ผลงานวิจัยนี้ไม่เคยเป็นส่วนหนึ่งในการอนุมัติปริญญาในระดับใดมาก่อน และไม่ได้ถูกใช้ในการยื่นขออนุมัติปริญญาในขณะนี้

ลงชื่อ.....
(นางสาวเจนจิรา หวังหลี)
นักศึกษา

ชื่อวิทยานิพนธ์ กลไกสำหรับความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส
ผู้เขียน นางสาวเจนจิรา หวังหลี
สาขาวิชา วิทยาการคอมพิวเตอร์
ปีการศึกษา 2556

บทคัดย่อ

ปัจจุบันนี้มีการนำเว็บเซอร์วิสมาประยุกต์ใช้งานทางด้านธุรกิจเพิ่มมากขึ้น และการเรียกใช้งานเว็บเซอร์วิสนั้นจะมีเอกสาร WSDL (Web Service Description Language) ซึ่งเป็นเอกสารอธิบายวิธีการเรียกใช้งานเว็บเซอร์วิส เอกสาร WSDL ที่ไม่ได้เข้ารหัสเป็นช่องโหว่ให้ผู้ไม่ประสงค์ดีโจมตีหรือดักจับเอกสาร WSDL นี้เพื่อขโมยข้อมูลได้ เพื่อทำให้เอกสาร WSDL มีความปลอดภัยในการนำไปใช้งานโดยทั่วไปจะเข้ารหัสเอกสาร WSDL โดยใช้มาตรฐานความปลอดภัยของ XML คือ XML Signature และ XML Encryption อย่างไรก็ตามการเข้ารหัสด้วยวิธีการนี้ใช้ระยะเวลาพอสมควร วิทยานิพนธ์นี้จึงเสนอกลไกสำหรับความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส โดยการเข้ารหัสเอกสาร WSDL เพียงบางส่วน ซึ่งประยุกต์ใช้วิทยาการเข้ารหัสลับและการย่อยข้อความ เพื่อทำให้เอกสาร WSDL มีความปลอดภัยและไม่ถูกปลอมแปลงก่อนส่งไปยังผู้ขอใช้บริการ และช่วยลดระยะเวลาในการเข้ารหัสและถอดรหัส ผลการทดลองแสดงให้เห็นว่ากลไกดังกล่าวสามารถลดระยะเวลาในการเข้ารหัสและถอดรหัสเอกสาร WSDL และสามารถป้องกันการโจมตีของเอกสาร WSDL ได้ ช่วยให้ผู้ขอใช้บริการได้รับเอกสาร WSDL ที่เป็นความลับและถูกต้องสมบูรณ์จากผู้ให้บริการในการนำไปใช้งาน

Thesis Title Mechanism for Securing WSDL Web Service
Author Miss. Janejira Wanglee
Major Program Computer Science
Academic Year 2013

ABSTRACT

At present, web services are widely applied in business. Since Web Service Description Language (WSDL) is used to describe the public interface to a specific web service, unencrypted WSDL document is possible to be vulnerable to attacks or traps from unauthorized users. Generally, a WSDL document is encrypted using XML security standards: XML Signature and XML Encryption. However, it takes times to encrypt and decrypt the WSDL document. Then, this thesis proposes a mechanism for securing WSDL document in web service called SWSDL which uses partial encryption in the WSDL document. Cryptography algorithm and message digest are applied to this mechanism for confidentiality and protecting message forgery in WSDL document. The experimental result shows that SWSDL can reduce the encryption and decryption time and protect the WSDL document from attackers. It helps a service requester to receive confidential WSDL document and integrity from a service provider.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จได้ด้วยความช่วยเหลือและสนับสนุนจากบุคคลหลายฝ่าย ผู้วิจัยรู้สึกซาบซึ้งและขอกราบขอบพระคุณอย่างสูง คือ

ผู้ช่วยศาสตราจารย์ ดร.ลัดดา ปรีชาวีรกุล อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่กรุณาให้คำปรึกษาแนะนำ และช่วยเหลือในการแก้ปัญหาต่างๆ ให้แก่ผู้วิจัยเสมอมา พร้อมทั้งตรวจทานและแก้ไขวิทยานิพนธ์ให้แก่ผู้วิจัย

ดร.นพมาศ ปักเข็ม ประธานกรรมการสอบวิทยานิพนธ์ ที่กรุณาช่วยตรวจทานและแก้ไขวิทยานิพนธ์ให้มีความสมบูรณ์

ผู้ช่วยศาสตราจารย์ ดร.ศิริรัตน์ วณิชโยบล กรรมการในการสอบวิทยานิพนธ์ที่กรุณาให้ความรู้และข้อเสนอแนะในการทำวิจัย รวมทั้งตรวจทานแก้ไขวิทยานิพนธ์

ผู้ช่วยศาสตราจารย์ ดร.วิภาดา เวทย์ประสิทธิ์ กรรมการในการสอบวิทยานิพนธ์ที่กรุณาให้ความรู้และข้อเสนอแนะในการทำวิจัย รวมทั้งตรวจทานแก้ไขวิทยานิพนธ์

อาจารย์ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยสงขลานครินทร์ทุกท่านที่ให้ความรู้ทางด้านวิชาการ ซึ่งสามารถนำความรู้นี้มาใช้ในการทำวิทยานิพนธ์

เจ้าหน้าที่ภาควิชาวิทยาการคอมพิวเตอร์ และเจ้าหน้าที่บัณฑิตวิทยาลัยทุกท่านที่ให้ความช่วยเหลือ และอำนวยความสะดวกเกี่ยวกับเอกสารต่างๆ

เพื่อนๆ พี่ๆ และน้องๆ ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ ที่ให้คำปรึกษา และช่วยเหลือในการทำวิทยานิพนธ์

คุณพ่อ พี่ชาย และพี่สาว ที่ให้การสนับสนุนคอยเป็นห่วงสุขภาพและให้กำลังใจแก่ผู้วิจัยมาโดยตลอด

ผู้วิจัยขอขอบคุณทุกท่านเป็นอย่างสูง ณ โอกาสนี้

เจนจิรา หวังหลี

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

รายการตาราง

รายการภาพประกอบ

รายการภาพประกอบ (ต่อ)

รายการภาพประกอบ (ต่อ)

บทที่ 1

บทนำ

ในยุคสารสนเทศ อินเทอร์เน็ตเข้ามามีบทบาทสำคัญสำหรับการทำธุรกรรมออนไลน์ การเผยแพร่ข้อมูลสารสนเทศจากองค์กรไปยังผู้รับบริการ ทำให้ผู้รับบริการทำธุรกรรมออนไลน์และรับข้อมูลได้สะดวกและง่ายมากขึ้น ซึ่งปัจจุบันได้นำเว็บเซอร์วิส (Web Service) มาประยุกต์ใช้งานเพื่อตอบสนองความต้องการทางด้านธุรกิจ การทำธุรกรรมออนไลน์และการเผยแพร่ข้อมูลสารสนเทศ ตัวอย่างเช่น กรมสรรพากรเปิดให้บริการเว็บเซอร์วิส ชื่อ “VATService” เพื่อให้บริการข้อมูลการตรวจสอบสถานะ และรายละเอียดผู้ประกอบการที่จดทะเบียนกับสรรพากรถูกต้อง และมีสิทธิ์ออกใบกำกับภาษีได้ถูกต้องตามกฎหมาย เป็นต้น

เว็บเซอร์วิสเป็นการนำแนวคิดของสถาปัตยกรรมเชิงบริการ (Service Orientation Architecture) มาประยุกต์ใช้งาน ซึ่งเว็บเซอร์วิสสนับสนุนการทำงานระหว่างกัน (Interoperation) โดยใช้ภาษา XML ในการรับส่งข้อมูลระหว่างผู้ให้บริการและผู้ใช้บริการผ่านเครือข่ายอินเทอร์เน็ต ในการใช้งานเว็บเซอร์วิส การเข้าถึงบริการของเว็บเซอร์วิสที่ฝั่งผู้ให้บริการ (Service Provider) จัดเตรียมไว้ ฝั่งผู้ใช้บริการ (Service Requester) จำเป็นต้องรู้รายละเอียดวิธีการติดต่อเรียกใช้บริการเว็บเซอร์วิสนั้น ซึ่งอธิบายในเอกสาร WSDL (Web Service Description Language) โดยทั่วไปการใช้งานเว็บเซอร์วิสไม่มีกลไกความปลอดภัยเพื่อเผยแพร่เอกสาร WSDL ผ่านเครือข่ายอินเทอร์เน็ตจากผู้ให้บริการไปยังผู้ใช้บริการ ทำให้ผู้ไม่ประสงค์ดีสามารถค้นหาเอกสาร WSDL ผ่านทางเว็บบราวเซอร์ โดยใช้โปรแกรมค้นหา (Search Engine) ในกูเกิ้ล (Google) ค้นหาเอกสาร WSDL ได้โดยใช้คำสั่ง “inurl:wsdl site:amazon.com” หรือคำสั่ง “inurl:asmx?wsdl” เป็นต้น (Shah, 2013) เพื่อดึงเอกสาร WSDL มาใช้งาน ทำให้เอกสาร WSDL อาจถูกโจมตีได้ โดยนำรายละเอียดข้อมูลในเอกสาร WSDL ไปใช้ในทางที่ผิด เช่น การเข้าถึงข้อมูลทางด้านการเงิน การเข้าถึงข้อมูลส่วนตัวของบุคคลที่ไม่มีสิทธิ์เข้าถึงข้อมูลนั้น เป็นต้น ทำให้เกิดปัญหาเกี่ยวกับการเปิดเผยรายละเอียดของข้อมูลในเอกสารที่เป็นความลับให้กับบุคคลอื่นทราบ หรืออาจมีการปลอมแปลงเอกสารจากต้นทางไปยังปลายทาง

แม้ว่าจะมีตัวอย่างงานวิจัยที่ป้องกันการโจมตีของเอกสาร WSDL โดยการจำกัดผู้ใช้ในการเข้าถึงเอกสาร WSDL และข้อมูลในเอกสาร WSDL ต้องไม่มีข้อมูลที่เป็นประโยชน์ต่อผู้โจมตี (Sidharth และ Liu, 2007) ซึ่งงานวิจัยนี้ เอกสาร WSDL ไม่มีการป้องกัน

หรือเข้ารหัส ดังนั้นหากผู้โจมตีดักจับเอกสาร WSDL นี้ได้สามารถนำเอกสาร WSDL นี้โจมตีเว็บเซอร์วิสได้

ต่อมาได้มีงานวิจัยที่นำความปลอดภัยพื้นฐานของ XML คือ XML Encryption และ XML Signature มาประยุกต์ใช้งานกับเอกสาร WSDL โดยนำมาประยุกต์ใช้งานร่วมกับโครงสร้างพื้นฐานกุญแจสาธารณะของ PKI (Public Key Infrastructure) ในการเข้ารหัสแท็กเอกสาร WSDL ทั้งหมด (Shahgholi และคณะ, 2011) อย่างไรก็ตาม วิธีการนี้ทำให้การเข้ารหัสเอกสาร WSDL ต้องใช้ระยะเวลานานในกระบวนการทำงานเข้ารหัสและถอดรหัส จึงได้มีงานวิจัยที่เข้ารหัสเอกสาร WSDL บางส่วนที่สำคัญ คือ การเข้ารหัสแท็ก <messge> และแท็ก <portType> ด้วย XML Encryption โดยใช้วิธีการเข้ารหัสลับแบบสมมาตรหรือแบบอสมมาตรขึ้นอยู่กับการนำเว็บเซอร์วิสไปใช้งาน (Mirtalebi และ Khayyambashi , 2011) และภายหลังได้พัฒนาโดยนำ XML Signature มาประยุกต์กับ XML Encryption โดยเข้ารหัสเพียงบางแท็ก คือ แท็ก <message> และแท็ก <portType> เพื่อให้เอกสาร WSDL ปลอดภัยมากขึ้น (Mirtalebi และ Khayyambashi, 2012) ถึงแม้ว่า XML จะมี XML Encryption และ XML Signature เป็นมาตรฐานรองรับความปลอดภัยในส่วนนี้แล้วก็ตาม แต่ด้วยขั้นตอนวิธีที่ยุ่งยากซับซ้อนและใช้ระยะเวลาพอสมควรในการเข้ารหัสและถอดรหัสข้อมูลเพียงบางส่วน และการเข้ารหัสแค่เพียงแท็ก <message> และแท็ก <portType> ไม่เพียงพอสำหรับทำให้เอกสาร WSDL มีความปลอดภัยอาจทำให้เกิดการโจมตีเอกสาร WSDL ได้อีก

วิทยานิพนธ์นี้จึงเสนอกลไกความปลอดภัยของเอกสาร WSDL เพื่อทำให้เอกสาร WSDL มีความปลอดภัยและถูกต้องสมบูรณ์เมื่อส่งผ่านเครือข่ายอินเทอร์เน็ตจากฝั่งผู้ให้บริการไปยังฝั่งผู้ใช้บริการ โดยเข้ารหัสและถอดรหัสเอกสาร WSDL เพียงบางส่วนที่สำคัญ คือแอททริบิวต์ภายในแท็ก <message> แท็ก <portType> แท็ก <service> และแท็ก <operation> ภายใต้แท็ก <binding> อีกทั้งยังลดระยะเวลาในการเข้ารหัสและถอดรหัสเอกสาร WSDL ในเว็บเซอร์วิส เพื่อทำให้การใช้งานเว็บเซอร์วิสมีความปลอดภัยและถูกต้องสมบูรณ์

1.1 วัตถุประสงค์

1.1.1 เพื่อศึกษา วิเคราะห์ และออกแบบวิธีการป้องกันความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิสให้มีความปลอดภัยในการนำไปใช้งาน

1.1.2 เพื่อพัฒนาวิธีการป้องกันความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิสให้มีความปลอดภัยในการนำไปใช้งาน

1.2 ขอบเขตการดำเนินงาน

1.2.1 วิเคราะห์และออกแบบกลไกการทำงานสำหรับความปลอดภัยของเอกสาร WSDL ดังต่อไปนี้

1) วิเคราะห์เอกสาร WSDL1.1

2) สกัดบางแท็กและแอททริบิวต์ภายในเอกสาร WSDL เพื่อเข้ารหัสและสามารถป้องกันการโจมตีของเอกสาร WSDL1.1 ได้

3) ออกแบบวิธีการเข้ารหัสเอกสาร WSDL1.1 สำหรับฝั่งผู้ให้บริการ

4) ออกแบบวิธีการถอดรหัสเอกสาร WSDL1.1 สำหรับฝั่งผู้ใช้บริการ

1.2.2 พัฒนาและทดสอบกลไกการทำงานสำหรับความปลอดภัยของเอกสาร WSDL ให้สามารถลดระยะเวลาในการทำงานในขั้นตอนการเข้ารหัสและถอดรหัสเอกสาร WSDL1.1 และเอกสาร WSDL1.1 ที่ได้มีความปลอดภัยในการนำไปใช้งาน ตามที่ได้ออกแบบไว้

1.2.3 ประเมินประสิทธิภาพของวิธีการที่นำเสนอเปรียบเทียบกับงานวิจัยก่อนหน้านี้

1.3 ขั้นตอนและระยะเวลาการดำเนินงาน

1.3.1 ขั้นตอนการดำเนินงาน

1) ศึกษางานวิจัยและเอกสารที่เกี่ยวข้อง
   1.1) เว็บเซอร์วิส
   1.2) การโจมตีเอกสาร WSDL
   1.3) วิทยาการเข้ารหัสลับ
   1.4) เทคโนโลยีอื่นๆ ที่เกี่ยวข้อง
2) ศึกษาเทคโนโลยีและเครื่องมือเกี่ยวกับงานวิจัย
3) วิเคราะห์และออกแบบระบบการทำงาน
4) พัฒนาระบบการทำงาน
5) ทดสอบและปรับปรุงการทำงาน
6) เขียนบทความวิจัยและเผยแพร่
7) จัดทำเอกสารวิทยานิพนธ์

1.3.2 ระยะเวลาดำเนินงาน

เมษายน 2556 – พฤษภาคม 2557

1.3.3 แผนการดำเนินงาน

ระยะเวลาการดำเนินงานวิจัยแสดงดังตารางที่ 1.1

ตารางที่ 1.1 ระยะเวลาการดำเนินงานวิจัย

กิจกรรม/ขั้นตอนการดำเนินงาน	เดือน													
	พ.ศ. 2556									พ.ศ. 2557				
	4	5	6	7	8	9	10	11	12	1	2	3	4	5
1. ศึกษางานวิจัยและเอกสารที่เกี่ยวข้อง	←	—	—	—	—	—	—	—	—	—	→			
2. ศึกษาเทคโนโลยีและเครื่องมือเกี่ยวกับงานวิจัย			←	—	—	→								
3. วิเคราะห์และออกแบบระบบการทำงาน					←	—	—	—	→					
4. พัฒนาระบบการทำงาน						←	—	—	—	→				
5. ทดสอบและปรับปรุงการทำงาน									←	—	→			
6. เขียนบทความวิจัยและเผยแพร่									←	—	—	—	→	
7. จัดทำเอกสารวิทยานิพนธ์					←	—	—	—	—	—	—	—	—	→

1.4 สถานที่และเครื่องมือที่ใช้

1.4.1 สถานที่

ห้องปฏิบัติการวิจัยเทคโนโลยีระบบสารสนเทศและการประยุกต์ (CS207) ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยสงขลานครินทร์

1.4.2 เครื่องมือที่ใช้

1) ด้านฮาร์ดแวร์

เครื่องคอมพิวเตอร์ส่วนบุคคล หน่วยประมวลผลกลางขนาด 2.27 GHz หน่วยความจำขนาด 4 GB และฮาร์ดดิสก์ความจุ 320 GB สำหรับพัฒนาและทดสอบระบบ

2) ด้านซอฟต์แวร์

2.1) ระบบปฏิบัติการ Windows 7 Ultimate Service Pack

2.2) ภาษาที่ใช้ในการพัฒนาระบบ Java

2.3) โปรแกรม Eclipse

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. ได้วิธีการป้องกันความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส สำหรับการนำเอกสาร WSDL ไปใช้งาน
2. ลดระยะเวลาที่ใช้ในการเข้ารหัส โดยเข้ารหัสเอกสารบางส่วน
3. ได้เอกสาร WSDL ที่ปลอดภัยและถูกต้องสมบูรณ์

บทที่ 2

ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในบทนี้จะกล่าวถึงทฤษฎีต่างๆ ที่ใช้ในการออกแบบและพัฒนากลไกสำหรับความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส ประกอบด้วย เว็บเซอร์วิส (Web Service) ภัยคุกคามเอกสาร WSDL (WSDL Threats) ความปลอดภัยของข้อมูล (Information Security) ความปลอดภัยของเว็บเซอร์วิส (Web Service Security) และงานวิจัยที่เกี่ยวข้องกับความปลอดภัยของเอกสาร WSDL

2.1 เว็บเซอร์วิส (Web Service)

เว็บเซอร์วิส (Cerami, 2002) เป็นส่วนหนึ่งของแนวคิดสถาปัตยกรรมเชิงบริการ (Service Orientation Architecture: SOA) เป็นบริการที่มีการใช้งานแบบแยกส่วน สามารถอธิบายรายละเอียด การลงทะเบียน และการสนับสนุนการทำงานระหว่างคอมพิวเตอร์ผ่านเครือข่าย โดยใช้ภาษา XML ในการรับส่งข้อมูลระหว่างผู้ให้บริการและผู้ใช้บริการ ซึ่งปัจจุบันนี้มีการนำเว็บเซอร์วิสมาใช้งานมากมาย ตัวอย่างเช่น Amazon Flexible Payment Service (FPS) เป็นเว็บเซอร์วิสเกี่ยวกับการทำธุรกรรมออนไลน์ของเว็บไซต์อะเมซอน (Coulouris และคณะ, 2012) และกรมสรรพากร (กรมสรรพากร, 2012: Online) เปิดให้บริการด้านเว็บเซอร์วิส โดยมีบริการ “CheckTINPINService” ให้บริการตรวจสอบความถูกต้องของเลขบัตรประจำตัวผู้เสียภาษีอากรและเลขบัตรประชาชน ว่าเป็นหมายเลขที่มีอยู่จริงและเป็นหมายเลขที่ถูกต้อง เป็นต้น

2.1.1 ภาษา XML

ภาษา XML (Extensible Markup Language) (Dykes และ Tittel, 2005 ;Fawcett และคณะ, 2012) ได้ถูกกำหนดให้เป็นมาตรฐานโดย W3C (World Wide Web Consortium) ซึ่งเป็นภาษา Markup ที่อธิบายถึงรายละเอียดของรูปแบบและโครงสร้างของข้อมูล จุดประสงค์หลักของ XML คือการแบ่งส่วนของข้อมูลเพื่อแสดงผล และมีต้นกำเนิดมาจากภาษา SGML (Standard Generalized Markup Language)

2.1.1.1 องค์ประกอบสำคัญของภาษา XML

1.1) แท็ก (Tag) และ อิลิเมนต์ (Element) ซึ่งเป็นส่วนประกอบสำคัญ ของ XML โดยที่ อิลิเมนต์ ประกอบด้วยแท็กเปิด แท็กปิดและข้อความภายในแท็ก

```
<name> Janejira </name>
```

ภาพประกอบ 2.1 ตัวอย่างอิลิเมนต์ของ XML

จากภาพประกอบ 2.1 แท็ก <name> คือแท็กเปิด และแท็ก </name> คือแท็กปิด และอิลิเมนต์หมายถึง <name> Janejira </name>

1.2) แอททริบิวต์ (Attribute) ซึ่งในแท็กเปิดของอิลิเมนต์สามารถมีแอททริบิวต์ได้ โดยแอททริบิวต์ประกอบด้วยชื่อและค่าของแอททริบิวต์นั้น ซึ่งภาพประกอบ 2.2 แสดงแอททริบิวต์ชื่อ “Type” ระบุค่าเท่ากับ “permanent”

```
<Customer Type=”permanent”> Janejira </Customer>
```

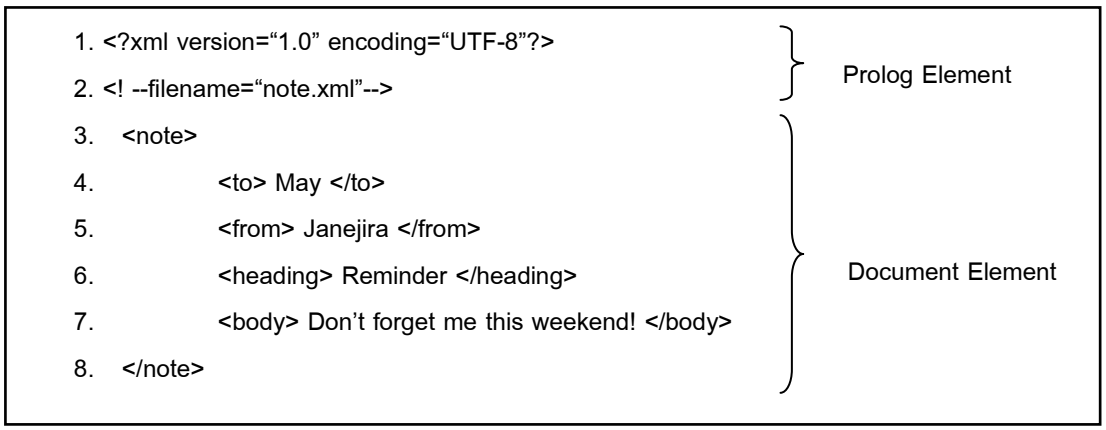
ภาพประกอบ 2.2 ตัวอย่างแอททริบิวต์ของ XML

2.1.1.2 โครงสร้างของภาษา XML

โครงสร้างของ XML แบ่งเป็น 2 ส่วนสำคัญ อธิบายในภาพประกอบ 2.3 ดังนี้

1) Prolog Element เป็นส่วนแรกของเอกสาร XML เป็น Optional (มีหรือไม่มีก็ได้) เป็นการประกาศเกี่ยวกับรูปแบบของ XML ประกอบด้วย รุ่นของเอกสาร XML และรูปแบบการเข้ารหัสเอกสาร XML โดยบรรทัดที่ 1 เป็นการระบุเวอร์ชั่นของ XML และระบุรูปแบบการเข้ารหัสตัวอักษร เช่น “UTF-8” และ บรรทัดที่ 2 ระบุ Comment

2) Document Element หรือ Root Element เป็นส่วนที่อธิบายถึงเนื้อหา ซึ่งประกอบด้วย อิลิเมนต์ย่อย โดยบรรทัดที่ 3 – 8 คือส่วนของ Document Element



ภาพประกอบ 2.3 ตัวอย่างโครงสร้าง XML

2.1.1.3 กฎพื้นฐานในการเขียน XML (Well-Formed)

คือการเขียนเอกสาร XML ให้มีโครงสร้างไวยากรณ์ถูกต้องตามกฎของ W3C

1) ทุกอิลิเมนต์ในเอกสาร XML จะต้องมีแท็กเปิดและแท็กปิดคู่กันเสมอ เช่น <name> Janejira </name>

2) XML มีลักษณะเป็น Case Sensitive คือตัวอักษรพิมพ์เล็กและตัวอักษรพิมพ์ใหญ่มีความหมายที่แตกต่างกัน เช่น <name> Janejira </name> และ <NAME> Jane </NAME>

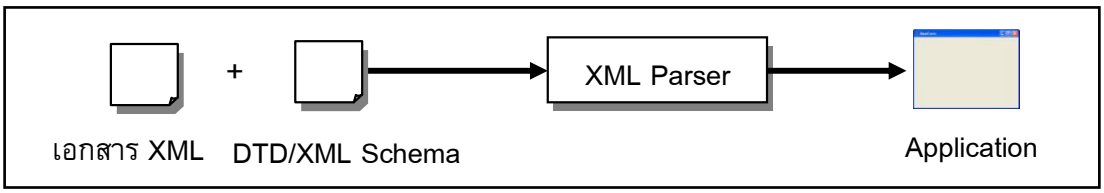
3) ห้ามเขียนแท็ก XML ซ้อนทับกัน (Overlap) คือ ไม่สามารถสลับตำแหน่งของแท็กได้ เช่น <Customer><name> Janejira </Customer></name>

4) เอกสาร XML ประกอบด้วย Root Element เพียงหนึ่งเท่านั้น

5) ค่าของแอททริบิวต์ในเอกสาร XML จะต้องอยู่ภายในเครื่องหมาย " " เช่น <Customer Type="permanent"> Janejira </Customer>

2.1.1.4 XML Parser

XML Parser คือตัวแปลเอกสาร XML ซึ่งเป็นตัวกลางระหว่างเอกสาร XML และโปรแกรมประยุกต์ในการเข้าถึงเอกสาร XML โดยการตรวจสอบความถูกต้องในการเขียนเอกสาร XML ตามกฎของ DTD หรือ XML Schema และวิเคราะห์โครงสร้างของเอกสาร XML



ภาพประกอบ 2.4 กระบวนการทำงานของ XML Parser

XML Parser แบ่งออกเป็น 2 ประเภทหลักๆ คือ

1) DOM (Document Object Model) เป็นการ Parser แบบโครงสร้างต้นไม้ (Tree-based) ซึ่งรูปแบบของเอกสาร XML เปรียบเสมือนโหนดของต้นไม้ โดยโปรแกรมประยุกต์สามารถที่จะค้นหาโหนด อ่านข้อมูล และปรับปรุงเนื้อหาของโหนดได้

2) SAX (Simple API for XML) เป็นการ Parser ขึ้นอยู่กับเหตุการณ์ (Event-driven)เอกสาร XML จะถูกวิเคราะห์โดยขึ้นอยู่กับแต่ละเหตุการณ์

การทำงานของ DOM และ SAX มีข้อดีและข้อด้อยต่างกันขึ้นอยู่ลักษณะงาน โดยสามารถเปรียบเทียบการทำงานระหว่าง DOM และ SAX (วิชุตา, 2552) ดังตารางที่ 2.1

ตารางที่ 2.1 เปรียบเทียบการทำงานระหว่าง DOM และ SAX

การทำงาน	DOM	SAX
การสำรวจข้อมูล	Tree-based Parser	Event-driven Parser
หน่วยความจำ	ใช้หน่วยความจำเยอะเพราะต้องโหลดเอกสารเข้ามาทั้งหมด	ใช้หน่วยความจำน้อยเพราะไม่ต้องโหลดเอกสารเข้ามาทั้งหมดเหมือน DOM
การเข้าถึงข้อมูล	สามารถเข้าถึงแบบสุ่ม (Random Access)ได้	เข้าถึงแบบ Sequential เท่านั้น
การอ่านข้อมูล	อ่านครั้งเดียวเพราะต้องอ่านเอกสารทั้งหมดเข้ามาเก็บไว้	อ่านข้อมูลทีละชุดขึ้นอยู่กับเหตุการณ์
การใช้งาน	เหมาะกับเอกสารขนาดเล็ก	เหมาะกับเอกสารขนาดใหญ่
การจัดการข้อมูล	อ่าน แก้ไข อิลิเมนต์ และเนื้อหาได้	อ่านได้อย่างเดียว
การเขียนโปรแกรม	เขียนโปรแกรมง่าย	เขียนโปรแกรมยาก

2.1.1.5 การนิยามโครงสร้างของเอกสาร XML

1) DTD (Document Type Definition) เป็นการระบุชนิดของข้อมูล XML ในเอกสาร ใช้อธิบายเนื้อหาเริ่มต้นที่ปรากฏขึ้นและวิธีการอ้างอิงภายใน XML ซึ่งมีองค์ประกอบหลัก ๆ ได้แก่ Elements, Attributes, Entities, PCDATA และ CDATA โดยการนิยามโครงสร้างขององค์ประกอบต่าง ๆ แสดงดังภาพประกอบ 2.5

note.dtd	test.xml
<!ELEMENT note (to, from, heading, body)> <!ELEMENT to (#PCDATA)> <!ELEMENT from (#PCDATA)> <!ELEMENT heading (#PCDATA)> <!ELEMENT body (#PCDATA)>	<?xml version="1.0"?> <!DOCTYPE note SYSTEM "note.dtd"> <note> <to> May </to> <from> Janejira </from> <heading> Reminder </heading> <body>Don't forget me this weekend!</body> </note>

ภาพประกอบ 2.5 ตัวอย่างการประกาศอิลิเมนต์

2) XML Schema ทำให้สามารถจัดการกับโครงสร้างเอกสาร XML ได้ดียิ่งขึ้น เนื่องจาก XML Schema ถูกเขียนด้วยภาษา XML จึงสามารถนำไปประยุกต์ใช้งานร่วมกับเทคโนโลยีที่เกี่ยวข้องกับ XML ได้ อีกทั้งยังรองรับการใช้งาน Namespace และชนิดข้อมูลได้มากขึ้นอีกด้วย แสดงดังภาพประกอบ 2.6

```
<xs:element name="note">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="to" type="xs:string"/>
            <xs:element name="from" type="xs:string"/>
            <xs:element name="heading" type="xs:string"/>
            <xs:element name="body" type="xs:string"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
```

ภาพประกอบ 2.6 ตัวอย่าง XML Schema

2.1.2 สถาปัตยกรรมของเว็บเซอร์วิส

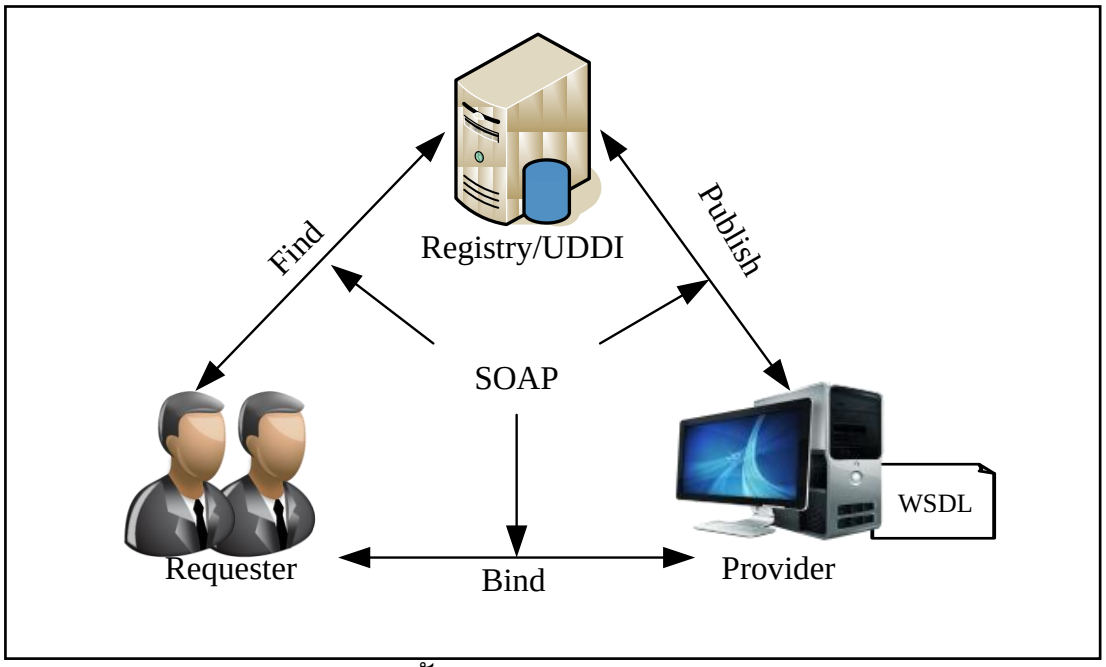
สถาปัตยกรรมของเว็บเซอร์วิสถูกแบ่งโดยบทบาทการดำเนินการกับเว็บเซอร์วิสเป็น 2 แบบ ดังนี้

2.1.2.1 การดำเนินการกับเว็บเซอร์วิส แบ่งเป็น 3 ส่วนสำคัญ อธิบายในภาพประกอบ 2.7 ดังนี้

1) ผู้ให้บริการ (Provider) เตรียมบริการเซอร์วิสให้เรียกใช้และเผยแพร่เซอร์วิสที่จัดทำขึ้นซึ่งก็คือเอกสาร WSDL ไปยัง Registry หรือ Universal Description, Discovery and Integration (UDDI)

2) บริการลงทะเบียน (Registry หรือ UDDI) ให้บริการลงทะเบียนเซอร์วิสและค้นหาเซอร์วิส เป็นบริการเก็บรวบรวมเซอร์วิสที่ผู้ให้บริการประกาศไว้

3) ผู้ขอใช้บริการ (Requester) เป็นผู้เรียกใช้เซอร์วิส โดยค้นหาเซอร์วิสจาก Registry แล้วเรียกใช้เซอร์วิสจากผู้ให้บริการ ซึ่งบางเซอร์วิสอาจจะไม่มีการลงทะเบียนเซอร์วิสไว้ ผู้ขอใช้บริการต้องติดต่อขอเรียกใช้เซอร์วิสนั้นจากผู้ให้บริการโดยตรง



ภาพประกอบ 2.7 สถาปัตยกรรมของเว็บเซอร์วิส (Tran, 2013: Online)

2.1.2.2 Web Service Protocol Stack

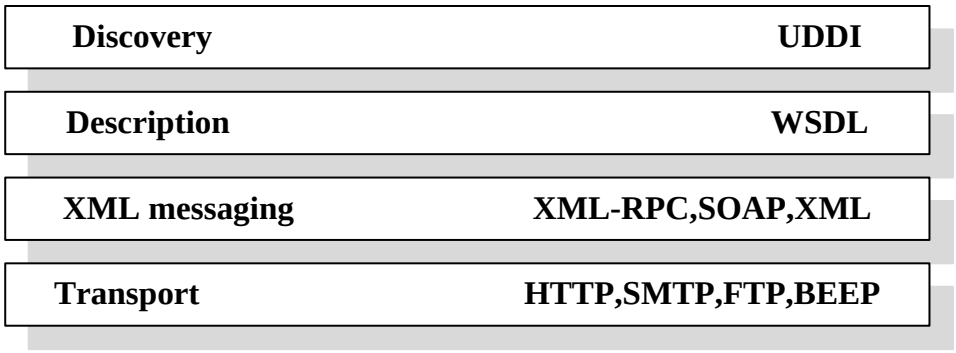
Web Service Protocol Stack เป็นอีกรูปแบบหนึ่งในการทำความเข้าใจสถาปัตยกรรมของเว็บเซอร์วิส โดยแบ่งเป็นลำดับชั้นได้ 4 ชั้นซึ่งแต่ละชั้นมีความสัมพันธ์กัน ดังอธิบายในภาพประกอบ 2.8

1) Service Transport เป็นการส่งผ่านข้อมูลระหว่างแอพพลิเคชัน ซึ่งในปัจจุบันโพรโทคอลที่อยู่บนชั้นนี้คือ HTTP SMTP FTP และโพรโทคอลใหม่ เช่น BEEP

2) XML Messaging เป็นการเข้ารหัสข้อความ (Encode) โดยทั่วไปอยู่ในรูปแบบของภาษา XML ประกอบด้วยโพรโทคอล XML-RPC และ SOAP

3) Service Description เป็นการอธิบายโครงสร้างทั่วไปของเว็บเซอร์วิส ปัจจุบันรายละเอียดของเซอร์วิสถูกกำหนดผ่าน WSDL

4) Service Discovery เป็นเซอร์วิสกลางในการลงทะเบียน และการเผยแพร่หรือค้นหาเซอร์วิสทั่วไป ปัจจุบันการค้นหาเซอร์วิสถูกกำหนดให้ค้นหาผ่าน UDDI



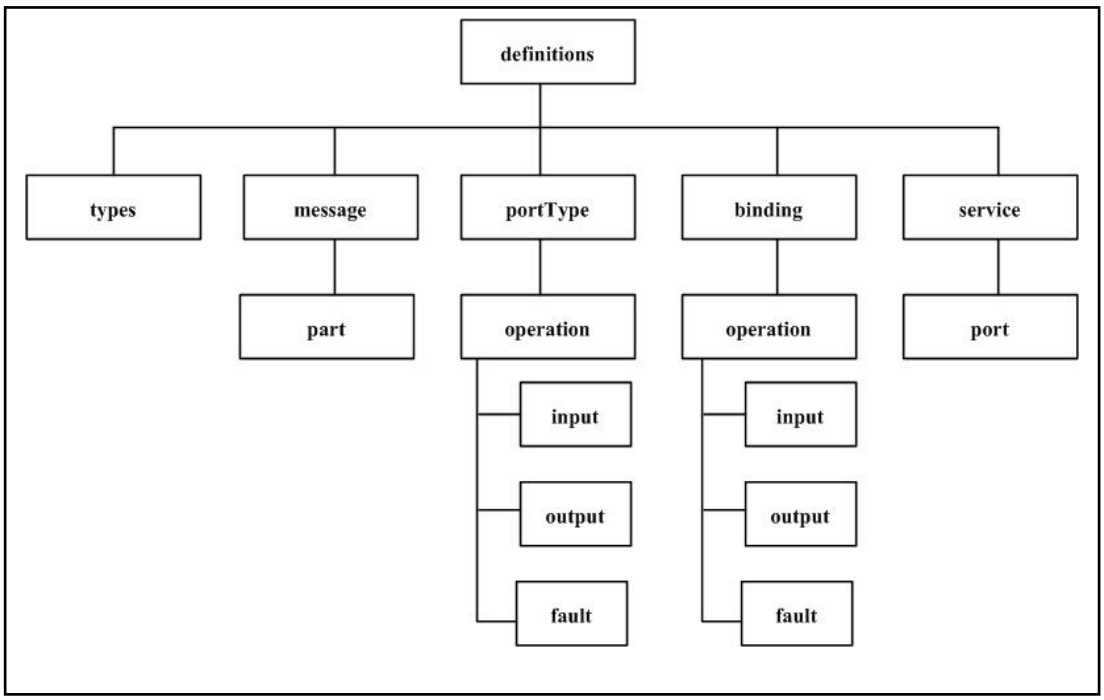
ภาพประกอบ 2.8 Web Service Protocol Stack (Cerami, 2002)

2.1.3 Web Service Description Language

Web Service Description Language หรือ WSDL (Newcomer, 2002) ถูกกำหนดโดยองค์กร W3C (World Wide Web Consortium) เป็นเอกสารที่มีรูปแบบของภาษา XML ซึ่งอธิบายเว็บเซอร์วิส โดยระบุรายละเอียดที่อธิบายการเรียกให้บริการเว็บเซอร์วิส กำหนดวิธีการติดต่อและตำแหน่งที่อยู่ของเว็บเซอร์วิส รุ่นที่ใช้งานอยู่ในปัจจุบันคือ 1.1 โครงสร้างของเอกสาร WSDL1.1 แสดงดังตารางที่ 2.2 และตัวอย่างแท็กพื้นฐานของเอกสาร WSDL1.1 แสดงดังภาพประกอบ 2.9

ตารางที่ 2.2 ตัวอย่างแท็กพื้นฐานของเอกสาร WSDL1.1

แท็ก	คำอธิบาย
<definitions>	กำหนดชื่อของเซอร์วิสและอธิบาย Namespace ที่ใช้ในเอกสาร WSDL
<types>	อธิบายชนิดของข้อมูล
<message>	อธิบายข้อมูลที่ใช้ในการแลกเปลี่ยนติดต่อระหว่างผู้ให้บริการและผู้ขอใช้บริการ
<portType>	กำหนดฟังก์ชันที่เรียกใช้งานและประกาศ Message เป็นข้อมูลเข้าและผลลัพธ์ของการทำงาน
<binding>	อธิบายโพรโทคอลและกำหนดรูปแบบการเข้ารหัสของข้อความ
<service>	กำหนดที่อยู่ของเว็บเซอร์วิส (URL)

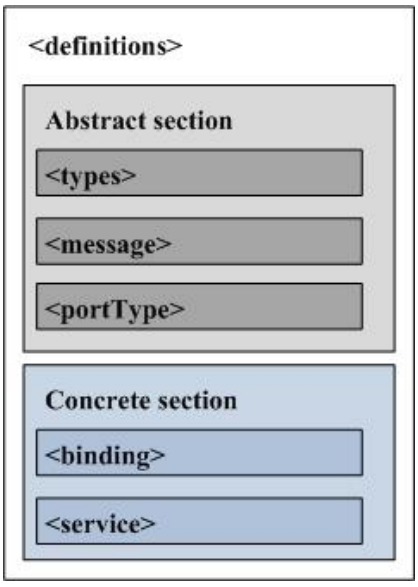


ภาพประกอบ 2.9 โครงสร้างแท็กของเอกสาร WSDL1.1

เอกสาร WSDL สามารถแบ่งได้เป็น 2 ส่วนดังนี้แสดงดังภาพประกอบ 2.10 คือ

1) ส่วนนามธรรม (Abstract Section) ระบุการทำงาน

2) ส่วนรูปธรรม (Concrete Section) ระบุโพรโทคอลและที่อยู่ของเซอร์วิสที่ถูกร้องขอ



ภาพประกอบ 2.10 โครงสร้างของ WSDL1.1 (Tran, 2013: Online)

```
<definitions targetNamespace="http://localhost/soap/HelloWorld">
    <types>
        <xsd:schema targetNamespace="http://localhost/soap/HelloWorld">
            <xsd:import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
            <xsd:import namespace="http://schemas.xmlsoap.org/wsdl/"/>          (1)
        </xsd:schema>
    </types>
    <message name="HelloWorldRequest">
        <part name="strName" type="xsd:string"/>
        <part name="strEmail" type="xsd:string"/>
    </message>                                                                 (2)
    <message name="HelloWorldResponse">
        <part name="return" type="xsd:string"/>
    </message>
    <portType name="HelloWorldPortType">
        <operation name="HelloWorld">
            <input message="tns:HelloWorldRequest"/>                           (3)
            <output message="tns:HelloWorldResponse"/>
        </operation>
    </portType>
```

ภาพประกอบ 2.11 (ก) ตัวอย่างส่วนนามธรรมของเอกสาร WSDL1.1

```
<binding name="HelloWorldBinding" type="tns:HelloWorldPortType">
    <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="HelloWorld">
            <soap:operation
        soapAction="http://localhost/php_webservice/webservice.php/HelloWorld"
            style="rpc"/>
        <input>
            <soap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
        </input>
        <output>
            <soap:body use="encoded" namespace=""
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
        </output>
    </operation>
</binding>
```

(4)

```
<service name="HelloWorld">
    <port name="HelloWorldPort" binding="tns:HelloWorldBinding">
        <soap:address location="http://localhost/php_webservice/webservice.php"/>
    </port>
</service>
</definitions>
```

(5)

ภาพประกอบ 2.11 (ข) ตัวอย่างส่วนรูปธรรมของเอกสาร WSDL1.1

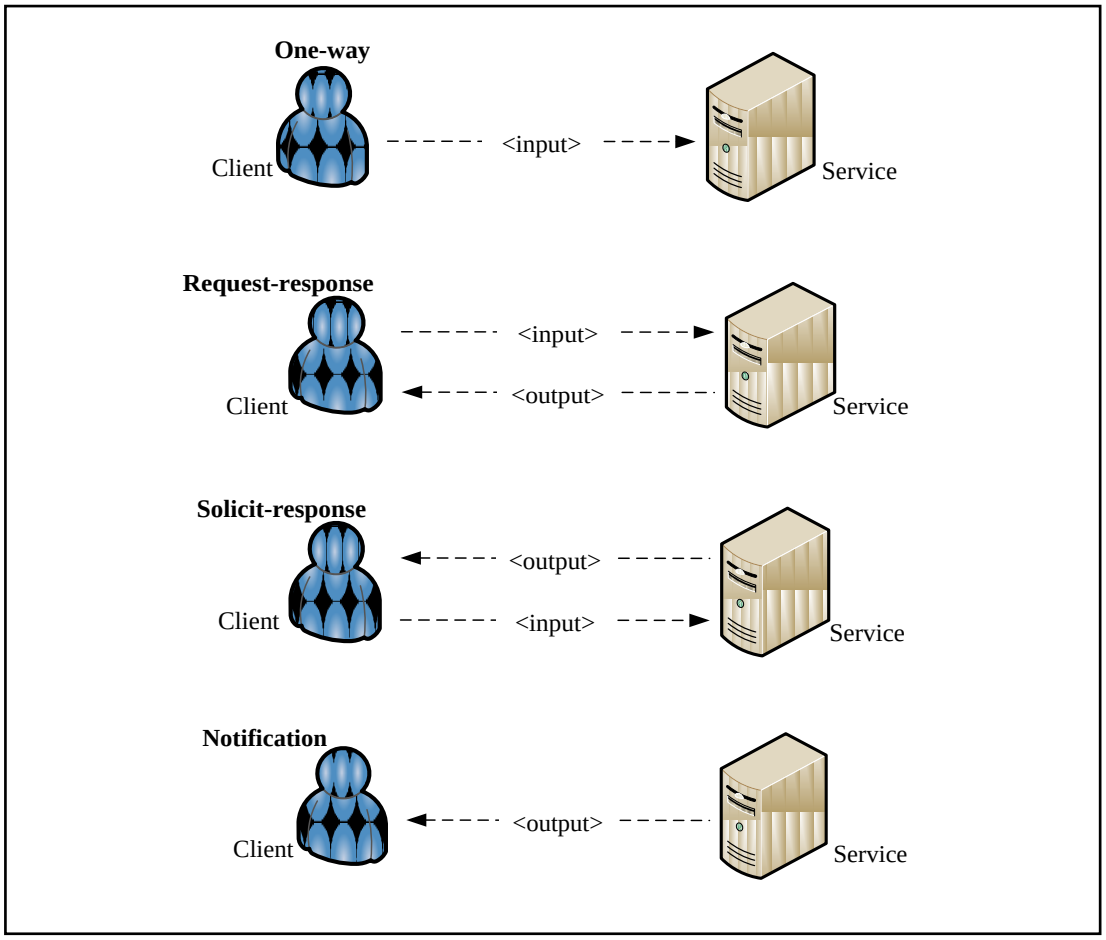
ภาพประกอบ 2.11 (ก) และ 2.11 (ข) แสดงตัวอย่างส่วนนามธรรมและรูปธรรมของเอกสาร WSDL1.1 ตามลำดับ ซึ่งมี แท็ก **<definitions>** บอกจุดเริ่มต้นของเอกสาร WSDL และมีแท็กย่อยภายในดังนี้

1) หมายเลข (1) คือส่วนของแท็ก **<types>** ประกอบด้วยคำจำกัดความของข้อมูลชนิดนามธรรม สำหรับแลกเปลี่ยนข้อความระหว่างต้นทางและปลายทาง

2) หมายเลข (2) คือส่วนแท็ก **<message>** สามารถมีได้หลายแท็กในเอกสาร WSDL ซึ่งประกอบด้วยแท็กต่าง ๆที่อยู่ภายในเพื่อบอกรายละเอียดข้อมูลแต่ละรายการ โดยมีแท็ก **<part>** เป็นส่วนระบุพารามิเตอร์ที่ใช้ในการติดต่อเซอร์วิส ภาพประกอบส่วนหมายเลข (2) ประกอบด้วย ข้อความร้องขอของแท็ก **<message>** ชื่อ “HelloWorldRequest”

ประกอบด้วยพารามิเตอร์ชื่อ “strName” และชื่อ “strEmail” มีชนิดข้อมูลเป็น “string” ข้อความตอบกลับชื่อ “HelloWorldResponse” ประกอบด้วยพารามิเตอร์ชื่อ “return” มีชนิดข้อมูลเป็น “string” ตามลำดับ

3) หมายเลข (3) คือส่วนแท็ก <portType> ประกอบด้วยแท็กย่อยคือแท็ก <operation> ระบุชื่อเซอร์วิสที่เปิดให้บริการ และภายในแท็กนี้มีแท็กย่อยคือ แท็ก <input> และแท็ก <output> เพื่อระบุข้อมูลเข้าและผลลัพธ์การทำงาน และอาจจะมีแท็ก <fault> ระบุรายละเอียดข้อผิดพลาดที่เกิดขึ้น ส่วนหมายเลข (3) การดำเนินการชื่อ “HelloWorld” ประกอบด้วยข้อมูลนำเข้าคือ “HelloWorldRequest” และผลลัพธ์การทำงานคือ “HelloWorldResponse” ซึ่งแท็ก <operation> มีประเภทการทำงานดังอธิบายในภาพประกอบ 2.12 และ ตารางที่ 2.3 ตามลำดับ



ภาพประกอบ 2.12 ประเภทการทำงานของแท็ก <opreation> (Cerami, 2002)

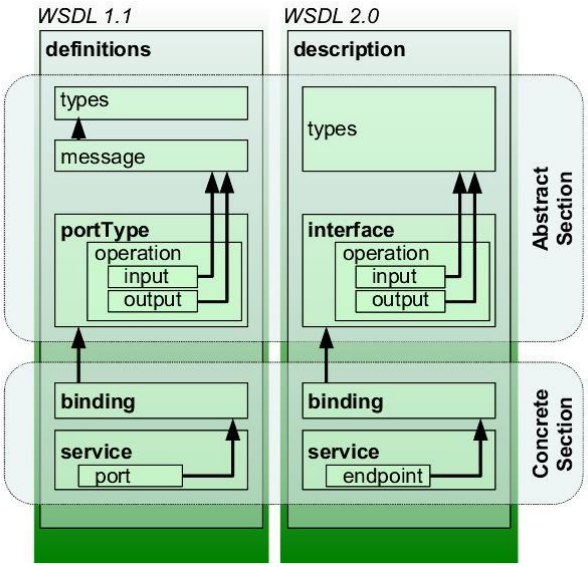
ตารางที่ 2.3 ประเภทการทำงานของแท็ก <operation>

ประเภทการทำงาน	คำอธิบาย
One-way	เซอร์วิสรับข้อมูลเข้ามาแต่ไม่มีผลลัพธ์กลับไป
Request-response	เซอร์วิสรับคำร้องขอเข้ามาและมีการส่งผลลัพธ์กลับไป
Solicit-response	เซอร์วิสส่งคำร้องขอออกไปและรอผลลัพธ์ตอบกลับ
Notification	เซอร์วิสส่งข้อมูลออกไปและไม่รอตอบกลับ

4) หมายเลข (4) คือแท็ก <binding> ประกอบด้วยแท็กย่อยภายในคือแท็ก <soap:binding> ระบุรูปแบบโดยรวมของ SOAP จากภาพประกอบ 2.11 (ข) เป็นรูปแบบแบบ “rpc” และแอททริบิวต์ชื่อ “transport” ระบุการส่งผ่านข้อความ SOAP โดยผ่านโพรโทคอล HTTP และแท็ก <soap:body> ระบุรายละเอียดข้อความนำเข้าและผลลัพธ์การทำงาน

5) หมายเลข (5) คือส่วนแท็ก <service> ใช้ระบุ URL (Uniform Resource Identifier) ในการเข้าถึงเซอร์วิส

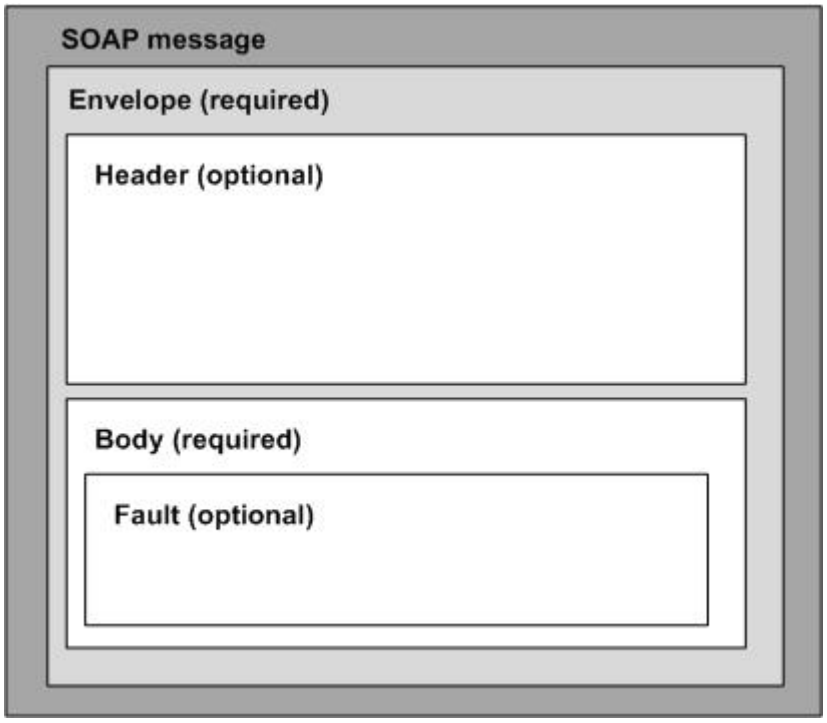
อย่างไรก็ตามในปัจจุบัน W3C ได้ระบุข้อกำหนดเกี่ยวกับ WSDL2.0 ซึ่งมีข้อแตกต่างจากเดิม โดยยกเลิกแท็ก <message> และอธิบายภายในแท็ก <types> เปลี่ยนชื่อแท็ก <portType> เป็นแท็ก <interface> และเปลี่ยนชื่อแท็ก <port> เป็นแท็ก <endpoint> ดังภาพประกอบ 2.13



ภาพประกอบ 2.13 เปรียบเทียบเอกสาร WSDL1.1 กับ WSDL2.0 (Wikipedia, 2014: Online)

2.1.4 Simple Object Access Protocol

Simple Object Access Protocol หรือ SOAP (Cerami, 2002) เป็นโพรโทคอลซึ่งมีรูปแบบของ XML ในการแลกเปลี่ยนข้อมูลระหว่างคอมพิวเตอร์ และ SOAP สามารถส่งผ่านโพรโทคอลในชั้น Transport ได้ จุดเริ่มต้น คือ SOAP ใช้ Remote Procudure Call (RPC) ส่งผ่าน HTTP ในชั้น Transport ดังนั้น SOAP ช่วยให้ผู้ใช้บริการเชื่อมต่อกับบริการระยะไกลและเรียกใช้ Method ระยะไกลสะดวก ซึ่งจากภาพประกอบ 2.14 แสดงโครงสร้างของข้อความ SOAP ดังนี้



ภาพประกอบ 2.14 โครงสร้างของข้อความ SOAP

1) Envelope ส่วนที่ใช้บรรจุเนื้อหาของเอกสารทั้งหมด

2) Header ส่วนเพิ่มเติมของเอกสาร SOAP ซึ่งจะมีหรือไม่มีก็ได้

3) Body ส่วนเรียกใช้งานเซอร์วิส และผลลัพธ์ที่ได้จากเซอร์วิส

4) Fault ส่วนระบุรายละเอียดข้อผิดพลาดที่เกิดขึ้น เช่น ผู้ใช้บริการเรียก Operation ชื่อ “ValidateCreditCard” แต่เว็บเซอร์วิสไม่มี Operation นี้จึงแสดงข้อผิดพลาดที่เกิดขึ้นในส่วนของ Fault แสดงดังภาพประกอบ 2.15

```
1.  <?xml version='1.0' encoding='UTF-8'?>
2.  <SOAP-ENV:Envelope
        xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
        xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
        xmlns:xsd="http://www.w3.org/1999/XMLSchema">
3.      <SOAP-ENV:Body>
4.      <SOAP-ENV:Fault>
5.          <faultcode xsi:type="xsd:string">SOAP-ENV:Client</faultcode>
6.          <faultstring xsi:type="xsd:string">
                Failed to locate method (ValidateCreditCard) in class
                (examplesCreditCard) at /usr/local/ActivePerl-5.6/lib/
                site_perl/5.6.0/SOAP/Lite.pm line 1555.
7.          </faultstring>
8.      </SOAP-ENV:Fault>
9.      </SOAP-ENV:Body>
10. </SOAP-ENV:Envelope>
```

ภาพประกอบ 2.15 ตัวอย่าง Fault ของ SOAP Response

การแลกเปลี่ยนข้อมูลในรูปแบบของ SOAP แบ่งเป็น 2 ประเภทคือ

1) Request-response (In-out) เมื่อผู้ส่งส่งคำร้องขอ (SOAP Request) จะมีการตอบสนอง (SOAP Response) กลับมายังผู้ส่ง

```
1.  <?xml version='1.0' encoding='UTF-8'?>
2.  <SOAP-ENV:Envelope
        xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema">
3.      <SOAP-ENV:Body>
4.          <ns1:getTemp xmlns:ns1="urn:xmethods-Temperature"
            SOAPENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
5.          <zipcode xsi:type="xsd:string">10016</zipcode>
6.          </ns1:getTemp>
7.      </SOAP-ENV:Body>
8.  </SOAP-ENV:Envelope>
```

ภาพประกอบ 2.16 ตัวอย่างข้อความ SOAP Request

ภาพประกอบ 2.16 ตัวอย่างของข้อความ SOAP ซึ่งเรียกใช้เว็บเซอร์วิสเพื่อสอบถามข้อมูลเกี่ยวกับราคาสินค้ารหัส “10016”

```
1.  <?xml version='1.0' encoding='UTF-8'?>
2.  <SOAP-ENV:Envelope
        xmlns:SOAP-ENV=http://schemas.xmlsoap.org/soap/envelope/
        xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
        xmlns:xsd="http://www.w3.org/2001/XMLSchema">
3.      <SOAP-ENV:Body>
4.          <ns1:getTempResponse
            xmlns:ns1="urn:xmethods-Temperature"
            SOAP- ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
5.          <return xsi:type="xsd:float">71.0</return>
6.          </ns1:getTempResponse>
7.      </SOAP-ENV:Body>
8.  </SOAP-ENV:Envelope>
```

ภาพประกอบ 2.17 ตัวอย่างข้อความ SOAP Response

ภาพประกอบ 2.17 แสดงตัวอย่างข้อความ SOAP Response ที่ถูกส่งกลับมาจากเว็บเซอร์วิส ซึ่งเป็นข้อมูลเกี่ยวกับราคาสินค้าของรหัสที่ต้องการสอบถาม

2) SOAP Request (In-only) เมื่อผู้ส่งส่งคำร้องขอ (SOAP Request) จะไม่สนใจว่ามีการตอบสนอง (SOAP Response) กลับมาหรือไม่ เช่น การแจ้งเตือน

2.1.5 Universal Description, Discovery and Integration

Universal Description, Discovery and Integration หรือ UDDI (Cerami, 2002) เป็นการกำหนดการอธิบาย ค้นหา และ แหล่งรวมเว็บเซอร์วิส ทำให้องค์กรสามารถเผยแพร่และค้นหาบริการเว็บเซอร์วิสได้ ซึ่ง UDDI ประกอบด้วย 2 ส่วนคือ

1) UDDI กำหนดการสร้างการกระจายไดเรกทอรีของธุรกิจและเว็บเซอร์วิส ข้อมูลถูกเก็บอยู่ในรูปแบบภาษา XML และกำหนดรายละเอียด API สำหรับค้นหาข้อมูลที่มีอยู่และลงทะเบียนข้อมูลใหม่

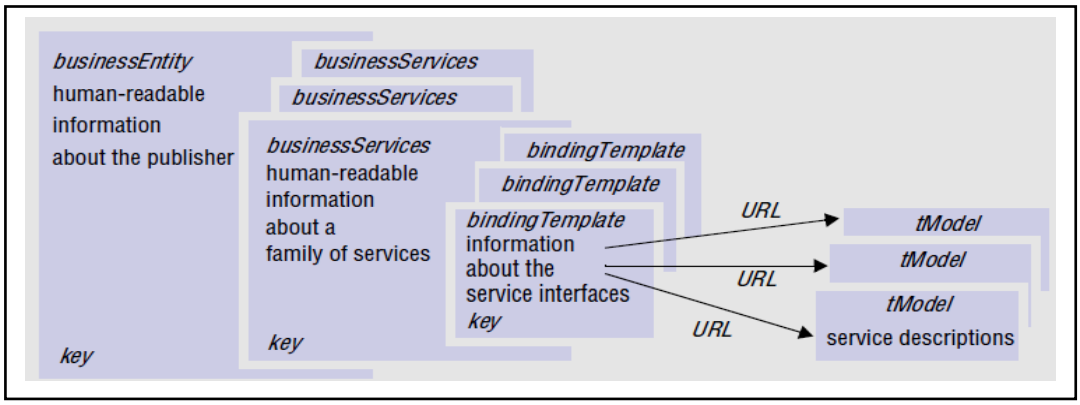
2) ฐานข้อมูลธุรกิจของ UDDI หรือ Cloud Services เป็นฐานข้อมูลสำหรับช่วยค้นหาข้อมูลที่มีอยู่ใน UDDI และช่วยให้บริษัทลงทะเบียนด้วยตัวเองได้

ข้อมูลภายใน UDDI แบ่งเป็น 3 ประเภทดังนี้

1) White Pages ข้อมูลทั่วไปเกี่ยวกับรายละเอียดของบริษัท ตัวอย่างเช่น ชื่อธุรกิจ รายละเอียดของธุรกิจ ข้อมูลการติดต่อ ที่อยู่และหมายเลขโทรศัพท์ เป็นต้น

2) Yellow Pages การจัดหมวดหมู่ข้อมูลทั่วไปของบริษัท หรือ บริการ ตัวอย่างเช่น ข้อมูลอุตสาหกรรม ผลิตภัณฑ์ รหัสทางภูมิศาสตร์ โดยแบ่งตามประเภทมาตรฐาน

3) Green Pages ข้อมูลเกี่ยวกับเว็บเซอร์วิส โดยทั่วไปรวมถึงข้อมูลภายนอกและที่อยู่สำหรับเรียกใช้เว็บเซอร์วิส UDDI ไม่จำกัดการอธิบายเว็บเซอร์วิสขึ้นอยู่กับ SOAP



ภาพประกอบ 2.18 แบบจำลองข้อมูลของ UDDI (Tran, 2013: Online)

จากภาพประกอบ 2.18 แบบจำลองข้อมูลของ UDDI ประกอบด้วย 4 ส่วน ดังนี้

1) BusinessEntity เป็นส่วนที่อธิบายองค์กรที่ให้บริการเว็บเซอร์วิส ประกอบด้วย ชื่อ คำอธิบายโดยย่อหรือวัตถุประสงค์ การติดต่อ และที่อยู่ขององค์กร

2) BusinessServices เป็นส่วนที่เก็บข้อมูลเกี่ยวกับเว็บเซอร์วิสและหมวดหมู่ของเว็บเซอร์วิส (BusinessEntity) แต่ละ BusinessServices ประกอบด้วย คำอธิบายของเซอร์วิส รายการของประเภทซึ่งอธิบายเซอร์วิส และรายการของ BindingTemplate ที่จะเชื่อมโยงไปยังข้อมูลของเซอร์วิสนั้น เช่น เป็นองค์กรเกี่ยวกับการท่องเที่ยว หรือขายหนังสือ เป็นต้น

3) BindingTemplate เป็นส่วนอธิบายข้อมูลทางเทคนิคสำหรับค้นหาเว็บเซอร์วิส BindingTemplate การเข้าถึงเว็บเซอร์วิส

4) tModel เป็นส่วนอธิบายต้นแบบเทคนิคแทนแนวคิดการนำกลับไปใช้ใหม่ เช่น ประเภทของเว็บเซอร์วิส โพรโทคอลที่ถูกใช้ในเว็บเซอร์วิส เป็นต้น เป็นส่วนระบุข้อมูลที่เก็บรายละเอียดของเว็บเซอร์วิส และอ้างอิงไปยังเอกสาร WSDL และเข้าถึงข้อมูลโดยใช้ URL

2.2 ภัยคุกคามเอกสาร WSDL (WSDL Threats)

ภัยคุกคามเว็บเซอร์วิส (Jensen และคณะ, 2009;Negm , 2004) แบ่งตามองค์ประกอบของเว็บเซอร์วิส ซึ่งจะประกอบด้วย การโจมตีในส่วนของ WSDL ดังนี้

2.2.1 WSDL Scanning (CAPEC, 2013: Online)

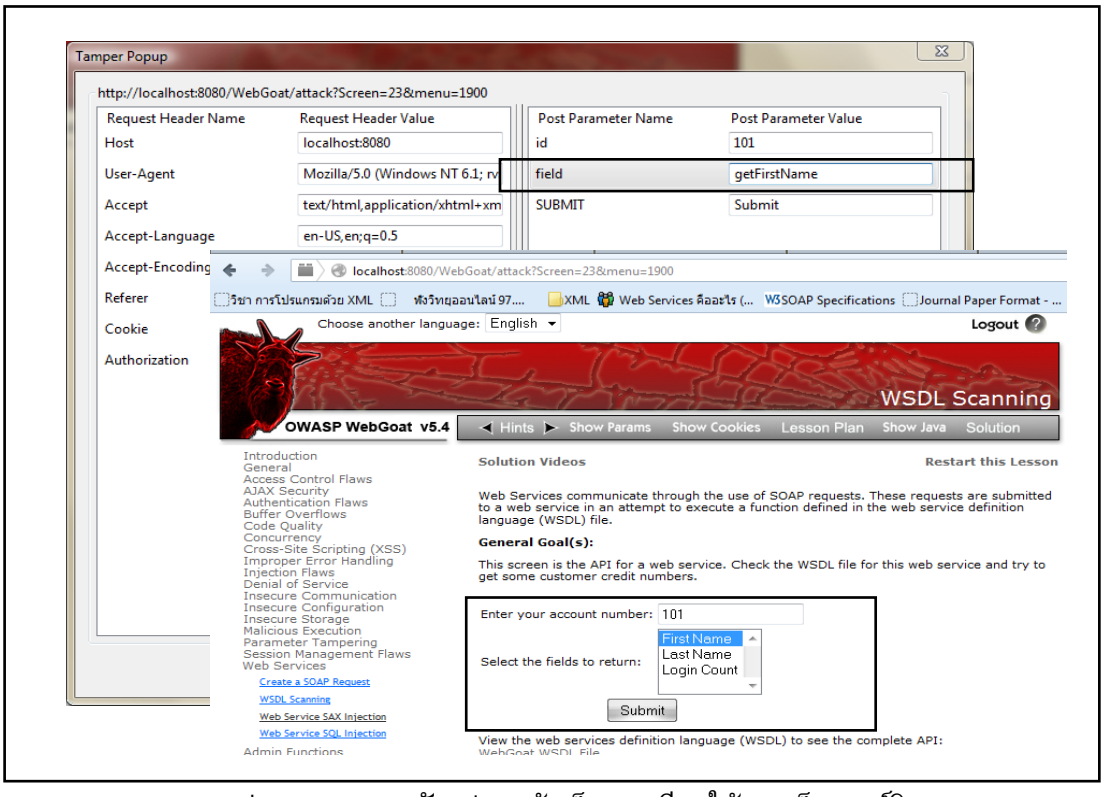
เอกสาร WSDL ประกอบด้วยข้อมูลการดำเนินการทุกอย่างของเว็บเซอร์วิสที่อธิบายไว้เพื่อให้ผู้ใช้งานสามารถเรียกใช้บริการนี้ได้ ดังนั้นผู้โจมตี (Hacker) สามารถที่จะวิเคราะห์เอกสาร WSDL นี้และดึงข้อมูลจากเว็บเซอร์วิส

ขั้นตอนการโจมตีเอกสาร WSDL

1) ผู้โจมตีอ่าน (Scan) เอกสาร WSDL ซึ่งเขียนด้วยภาษา XML โดยเอกสาร WSDL อธิบายรายละเอียดฟังก์ชัน วัตถุประสงค์การใช้งาน และ ชนิดของข้อมูล เป็นต้น ซึ่งเป็นประโยชน์มากสำหรับผู้โจมตี

2) ผู้โจมตีวิเคราะห์เอกสาร WSDL และพยายามหาช่องโหว่ โดยการส่งข้อความที่ตรงกับรูปแบบที่อธิบายไว้ในเอกสาร WSDL จนกว่าจะเจอช่องโหว่

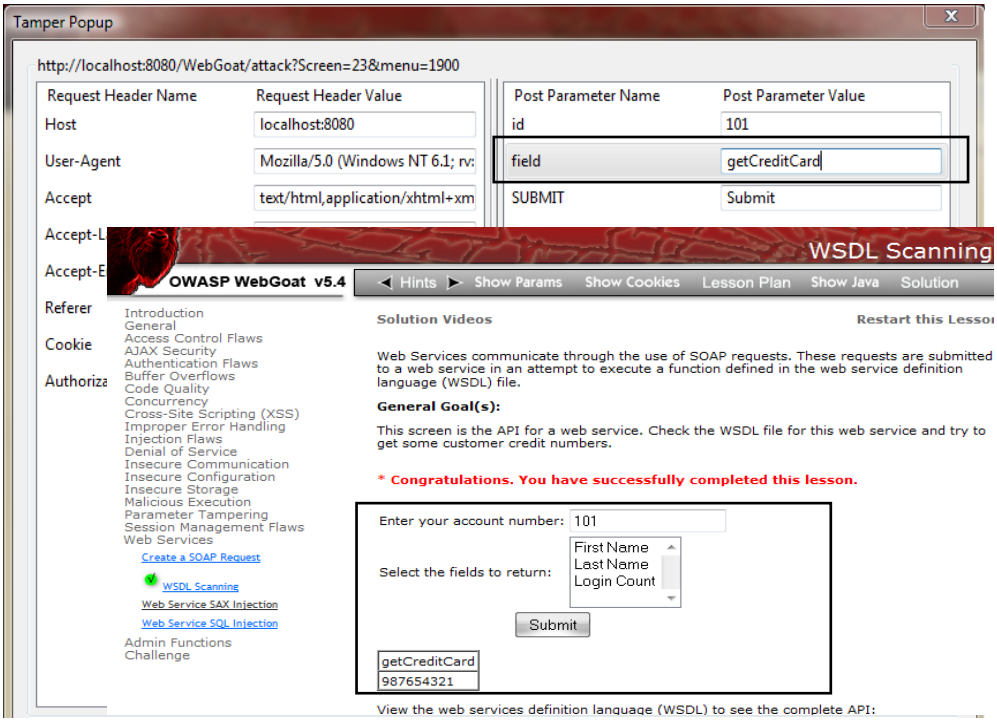
3) เมื่อผู้โจมตีเจอช่องโหว่ ผู้โจมตีส่งเนื้อหาที่ไม่พึงประสงค์ไปยังระบบ



ภาพประกอบ 2.19 ตัวอย่างหน้าเว็บเพจเรียกใช้งานเว็บเซอร์วิส

```
1.  <wsdl:portType name="WSDLScanning">
2.      <wsdl:operation name="getFirstName" parameterOrder="id">
3.          <wsdl:input message="impl:getFirstNameRequest" name="getFirstNameRequest"/>
4.          <wsdl:output message="impl:getFirstNameResponse"name="getFirstNameResponse"/>
5.      </wsdl:operation>
6.      <wsdl:operation name="getLastName" parameterOrder="id">
7.          <wsdl:input message="impl:getLastNameRequest" name="getLastNameRequest"/>
8.          <wsdl:output message="impl:getLastNameResponse" name="getLastNameResponse"/>
9.      </wsdl:operation>
10.     <wsdl:operation name="getCreditCard" parameterOrder="id">
11.         <wsdl:input message="impl:getCreditCardRequest" name="getCreditCardRequest"/>
12.         <wsdl:output message="impl:getCreditCardResponse" name="getCreditCardResponse"/>
13.     </wsdl:operation>
14.     <wsdl:operation name="getLoginCount" parameterOrder="id">
15.         <wsdl:input message="impl:getLoginCountRequest" name="getLoginCountRequest"/>
16.         <wsdl:output message="impl:getLoginCountResponse" name="getLoginCountResponse"/>
17.     </wsdl:operation>
18. </wsdl:portType>
```

ภาพประกอบ 2.20 ตัวอย่างแท็ก <portType> ของเอกสาร WSDL



ภาพประกอบ 2.21 ตัวอย่างการโจมตี WSDL Scanning

จากภาพประกอบ 2.19 - 2.21 แสดงตัวอย่างการโจมตีเอกสาร WSDL ตามลำดับ เป็นการจำลองการโจมตี WSDL Scanning ของ WebGoat (OWASP, 2013: Online) ภาพประกอบ 2.19 ตัวอย่างหน้าเว็บเพจเรียกใช้งานเว็บเซอร์วิส เป็นหน้าเพจสำหรับให้ผู้ใช้งานเรียกดู ชื่อ นามสกุล หรือจำนวนครั้งในการเข้าระบบได้จากหน้าเว็บเพจตามหมายเลขบัญชีผู้ใช้ ซึ่งผู้ไม่ประสงค์ดีหรือผู้โจมตี (Hacker) สามารถดึงข้อมูลสำคัญของเว็บเพจนี้ได้เมื่อผู้โจมตีเข้าถึงเอกสาร WSDL ต้นฉบับ ซึ่งเอกสาร WSDL ต้นฉบับแสดงในภาพประกอบ 2.20 เอกสาร WSDL ประกอบด้วยเซอร์วิสที่ให้บริการ 4 เซอร์วิส คือ “getFirstName” “getLastName” “getCreditCard” และ “getLoginCount” เมื่อผู้โจมตีเข้าถึงเอกสาร WSDL ต้นฉบับจะอ่านและวิเคราะห์เอกสาร WSDL ต้นฉบับประกอบด้วยเซอร์วิสที่เรียกใช้งานอะไรบ้าง ซึ่งจากตัวอย่างมีเซอร์วิส “getCreditCard” ประกอบอยู่ภายในเอกสาร WSDL ดังนั้นผู้โจมตีสามารถใช้ Plug-in ในเว็บบราวเซอร์ (Browser) คือ Tamper Data จากภาพประกอบ 2.19 และ 2.21 เมื่อผู้โจมตีป้อนหมายเลขบัญชีผู้ใช้เพื่อต้องการข้อมูลชื่อผู้ใช้เซอร์วิส ชื่อ “getFirstName” ในเว็บเซอร์วิสจะถูกเรียกใช้งานและแสดงใน Tamper Data ในภาพประกอบ 2.19 ผู้โจมตีเปลี่ยนชื่อเซอร์วิสใน Tamper Data จาก “getFirstName” เป็น “getCreditCard” เพื่อเรียกใช้งานเซอร์วิสนี้ ซึ่งจะแสดงหมายเลขบัตรเครดิต ดังภาพประกอบ 2.21 ตามลำดับ

2.2.2 Parameter Tampering

เป็นการโจมตีโดยการเปลี่ยนแปลงเนื้อหาภายในเอกสาร WSDL หรือ WSDL Spoofing (WS-Attacks.org, 2010: Online) แสดงดังภาพประกอบ 2.22 และ 2.23 ตามลำดับ

```
1.  <wsdl:definitions>
2.  <!-- wsdl:service names a new service "StockQuoteService" -->
3.  <wsdl:service name="StockQuoteService">
4.      <wsdl:documentation>My first service</wsdl:documentation>
5.      <!-- connect it to the binding "StockQuoteBinding" above -->
6.      <wsdl:port name="StockQuotePort"  binding="tns:StockQuoteBinding">
7.          <!-- give the binding an network address -->
8.          <soap:address location="http://example.com/stockquote"/>
9.      </wsdl:port>
10.   </wsdl:service>
11. </wsdl:definitions>
```

ภาพประกอบ 2.22 ตัวอย่างแท็ก <service> ของเอกสาร WSDL

```
1.  <wsdl:definitions>
2.  <!-- wsdl:service names a new service "StockQuoteService" -->
3.      <wsdl:service name="StockQuoteService">
4.          <wsdl:documentation>My first service</wsdl:documentation>
5.          <!-- connect it to the binding "StockQuoteBinding" above -->
6.          <wsdl:port name="StockQuotePort" binding="tns:StockQuoteBinding">
7.          <!-- give the binding an network address -->
8.   <!-- THE LOCATION WAS CHANGED FROM examples.com to ATTACK.com -->
9.              <soap:address location="http://ATTACK.com/stockquote"/>
10.         </wsdl:port>
11.     </wsdl:service>
12. </wsdl:definitions>
```

ภาพประกอบ 2.23 ตัวอย่างการโจมตี Parameter Tampering

จากภาพประกอบ 2.22 เป็นส่วนแท็ก <service> ในเอกสาร WSDL ผู้ไม่ประสงค์ดีปลอมแปลงเอกสาร WSDL ในแท็กนี้ โดยผู้ไม่ประสงค์ดีเปลี่ยน URL จาก "http://example.com/stockquote" เป็น “http://ATTACK.com/stockquote” ดังภาพประกอบ 2.23 เพื่อให้ผู้ใช้งานเว็บเซอร์วิสเรียกไปยัง URL ที่ผู้ไม่ประสงค์ดีกำหนดไว้ ทำให้ผู้ไม่ประสงค์ดีสามารถที่จะขโมยข้อมูลสำคัญได้

2.3 ความปลอดภัยของข้อมูล (Information Security)

ความปลอดภัยของข้อมูล (Rosenberg และ Remy, 2004) เป็นประเด็นสำคัญในการแลกเปลี่ยนข้อมูลระหว่างผู้รับกับผู้ส่งผ่านระบบอินเทอร์เน็ต ซึ่งอาจมีผู้ที่ประสงค์ร้ายต่อข้อมูลและต่อการได้รับข้อมูลเพื่อมาใช้ประโยชน์โดยที่ตนเองไม่สิทธิ์ในการเข้าถึงข้อมูลนั้น การที่จะทำให้ข้อมูลเกิดความปลอดภัยและเพื่อให้มั่นใจว่าข้อมูลจะไม่ถูกทำลาย เปลี่ยนแปลง หรือเข้าถึงได้โดยบุคคลที่ไม่เกี่ยวข้อง จำเป็นที่จะต้องมีองค์ประกอบที่ทำให้ข้อมูลมีความปลอดภัยดังนี้

1) การรักษาความลับ (Confidentiality) เป็นวิธีในการปกปิดข้อมูลให้เป็นความลับอยู่เสมอ โดยผู้ที่มีสิทธิ์ในการเข้าถึงเท่านั้นที่สามารถทราบรายละเอียดหรือเนื้อหาภายในได้

2) การมีบูรณภาพ (Integrity) เป็นวิธีที่รับรองว่าข้อมูลที่ถูกส่งจะไม่ถูกแก้ไขระหว่างการส่ง ซึ่งผู้รับจะได้รับข้อมูลที่ถูกต้องและสมบูรณ์

3) การพิสูจน์ตัวจริง (Authentication) เป็นวิธีในการตรวจสอบว่าเป็นบุคคลที่กล่าวอ้างจริง เพื่อยืนยันตัวบุคคลในการเข้าถึงข้อมูล

4) การไม่ปฏิเสธการกระทำ (Non-repudiation) เป็นวิธีที่ทำให้ผู้ส่งข้อมูลไม่สามารถปฏิเสธได้ว่าเป็นผู้ส่งข้อมูลนั้นออกไป

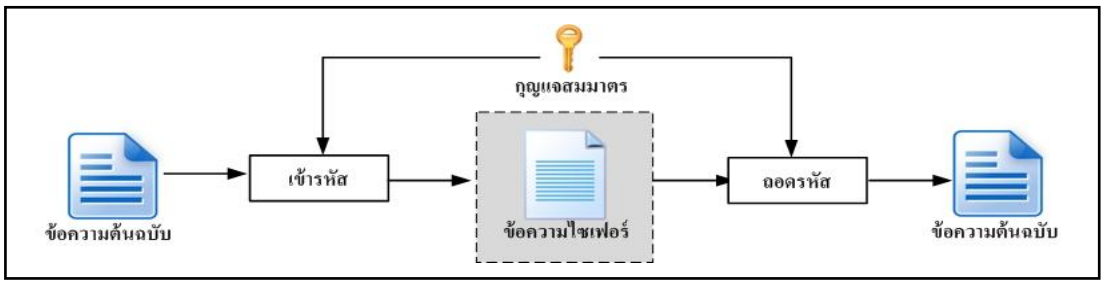
2.3.1 วิทยาการเข้ารหัสลับ (Cryptography)

องค์ประกอบที่ทำให้ข้อมูลมีความปลอดภัย เพื่อป้องกันไม่ให้ผู้ที่ไม่เกี่ยวข้องสามารถนำข้อมูลไปใช้ประโยชน์ได้ จึงต้องทำให้ข้อมูลเป็นความลับโดยการเข้ารหัสข้อมูลนั้น ซึ่งเรียกว่า วิทยาการเข้ารหัสลับ (Cryptography) (Nordbotten, 2009) ซึ่งมีองค์ประกอบที่สำคัญคือ ขั้นตอนวิธี (Algorithm) และกุญแจ (Key) ที่ใช้ในการเข้ารหัสและถอดรหัส โดยมีกระบวนการเข้ารหัสและถอดรหัสข้อความดังภาพประกอบ 2.24

ภาพประกอบ 2.24 กระบวนการเข้ารหัสและถอดรหัสข้อความ

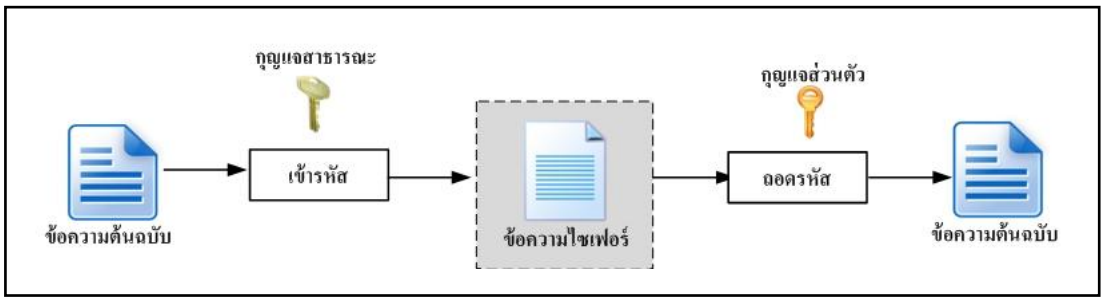
ภาพประกอบ 2.24 ผู้ส่งเข้ารหัสข้อความต้นฉบับด้วยขั้นตอนวิธีและกุญแจได้ข้อความลับหรือข้อความไซเฟอร์และส่งไปยังผู้รับ จากนั้นผู้รับถอดรหัสข้อความไซเฟอร์ด้วยขั้นตอนวิธีและกุญแจได้ข้อความต้นฉบับเดิม วิทยาการเข้ารหัสลับแบ่งได้เป็น 2 ประเภท ดังนี้

1. การเข้ารหัสลับแบบกุญแจสมมาตร (Symmetric Key Cryptography) หรือ การเข้ารหัสลับแบบกุญแจลับ (Secret Key Cryptography) โดยวิธีการนี้ใช้กุญแจตัวเดียวกันในการเข้ารหัสและถอดรหัสข้อความ ตัวอย่างการเข้ารหัสวิธีการนี้ได้แก่ AES (Advanced Encryption Standard) เป็นต้น ซึ่งกระบวนการเข้ารหัสและถอดรหัสข้อความโดยใช้กุญแจสมมาตร ดังภาพประกอบ 2.25



ภาพประกอบ 2.25 กระบวนการเข้ารหัสและถอดรหัสข้อความโดยใช้กุญแจสมมาตร

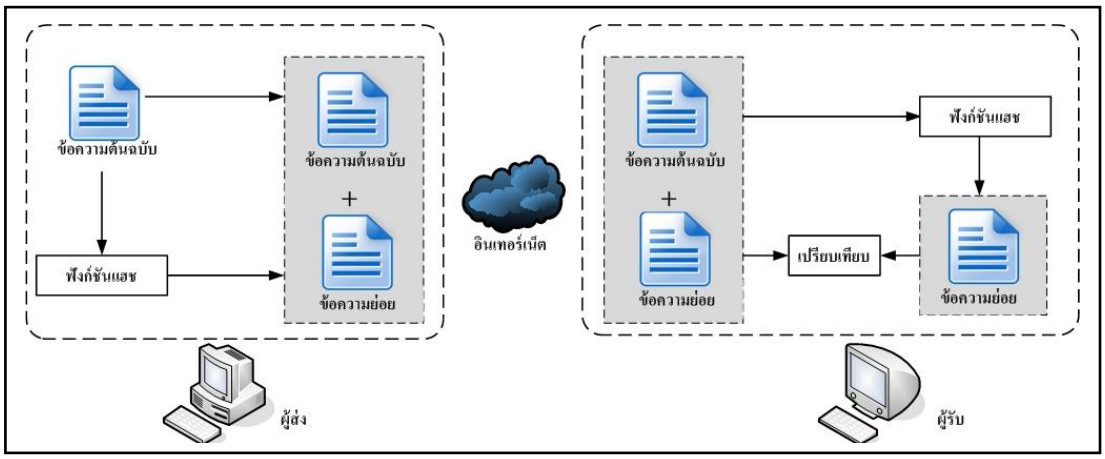
2. การเข้ารหัสลับแบบกุญแจอสมมาตร (Asymmetric Key Cryptography) หรือ การเข้ารหัสลับแบบกุญแจสาธารณะ (Public Key Cryptography) โดยวิธีการนี้ใช้กุญแจ 2 ตัว คือ กุญแจสาธารณะ (Public Key) และกุญแจส่วนตัว (Private Key) ซึ่งกุญแจสาธารณะประกาศให้บุคคลทั่วไปทราบได้ แต่กุญแจส่วนตัวจะถูกเก็บเป็นความลับ ตัวอย่างการเข้ารหัสวิธีการนี้ ได้แก่ RSA (Rivest, Shamir and Adleman) และการลงลายมือชื่อดิจิทัล DSA (Digital Signature Algorithm) เป็นต้น ซึ่งกระบวนการเข้ารหัสและถอดรหัสข้อความโดยใช้กุญแจอสมมาตร ดังภาพประกอบ 2.26



ภาพประกอบ 2.26 กระบวนการเข้ารหัสและถอดรหัสข้อความโดยใช้กุญแจอสมมาตร

2.3.2 การย่อยข้อความ

องค์ประกอบที่ทำให้ข้อมูลมีความปลอดภัย เพื่อให้แน่ใจว่าข้อความนั้นไม่ถูกแก้ไขระหว่างการส่งจากต้นทางไปยังปลายทาง คือ การมีบูรณภาพ (Integrity) ของข้อความโดยใช้ฟังก์ชันแฮช (Hash Function) (Mogollon, 2007) ซึ่งเป็นฟังก์ชันในการย่อยข้อความตัวอย่างขั้นตอนวิธีนี้ได้แก่ Message Digest 5 (MD5) และ Secure Hash Standards (SHA-1, SHA-256, SHA-384, และ SHA-512) เป็นต้น ซึ่งกระบวนการตรวจสอบการมีบูรณภาพของข้อความโดยใช้ฟังก์ชันแฮช ดังภาพประกอบ 2.27



ภาพประกอบ 2.27 กระบวนการตรวจสอบการมีบูรณภาพ

จากภาพประกอบ 2.27 กระบวนการตรวจสอบการมีบูรณภาพ อธิบายได้ดังนี้

1) ฝั่งผู้ส่งนำข้อความต้นฉบับมาผ่านฟังก์ชันแฮชได้เป็นข้อความย่อย และส่งข้อความต้นฉบับและข้อความย่อยให้กับฝั่งผู้รับ

2) ฝั่งผู้รับย่อยข้อความต้นฉบับด้วยฟังก์ชันแฮช ได้ข้อความย่อยแล้วนำมาเปรียบเทียบกับข้อความย่อยที่ได้รับมาจากฝั่งผู้ส่ง ถ้าผลลัพธ์ที่ได้เหมือนกัน แสดงว่าข้อความที่ส่งมาไม่ถูกแก้ไขหรือเปลี่ยนแปลงระหว่างการส่ง แต่ถ้าผลลัพธ์ที่ได้ไม่เหมือนกันแสดงว่าข้อความที่ส่งมาถูกแก้ไขหรือเปลี่ยนแปลงระหว่างการส่ง

2.3.3 อัลกอริทึมเข้ารหัสลับและการย่อยข้อความ

1) อัลกอริทึม AES

Advanced Encryption Standard หรือ AES (Mogollon, 2007) เป็นอัลกอริทึมเข้ารหัสแบบกุญแจสมมาตร ได้รับการพัฒนาโดย Joan Daemen และ Vincent Rijmen ในปี ค.ศ. 2001 อัลกอริทึมนี้ได้รับการคัดเลือกโดยหน่วยงาน National Institute of Standards and Technology (NIST) ของสหรัฐอเมริกาให้เป็นมาตรฐานในการเข้ารหัส อัลกอริทึมมีความเร็วสูงและมีขนาดกะทัดรัดซึ่งเข้ารหัสแบบบล็อกไซเฟอร์ (Block Cipher) และขนาดของบล็อคเป็น 128 บิต โดยใช้กุญแจขนาด 128 192 และ 256 บิต จำนวนรอบการทำงานขึ้นอยู่กับขนาดของบล็อคและขนาดของกุญแจ แสดงดังตารางที่ 2.4

ตารางที่ 2.4 จำนวนรอบการทำงาน

ขนาดของบล็อก (บิต)	ขนาดของกุญแจ (บิต)		
	128	192	256
128	10 รอบ	12 รอบ	14 รอบ
192	12 รอบ	12 รอบ	14 รอบ
256	14 รอบ	14 รอบ	14 รอบ

ข้อดี

- มีประสิทธิภาพดีในด้านของฮาร์ดแวร์และซอฟต์แวร์
- ใช้หน่วยความจำต่ำซึ่งเหมาะกับสภาพแวดล้อมที่มีพื้นที่จำกัด
- มีความยืดหยุ่นในด้านขนาดบล็อกและขนาดของกุญแจและรองรับการเปลี่ยนแปลงในจำนวนรอบการทำงานได้

```
Cipher(byte in[4*Nb],byte out[4*Nb],word w[Nb*(Nr+1)])
Begin
        Byte state[4,Nb]
        State = in
        AddRoundKey(state, w[0,Nb-1])
        For round = 1 step 1 to Nr-1
                SubBytes(state)
                ShiftRows(state)
                MixColumns(state)
                AddRoundKey(state, w[round*Nb, (round+1)*Nb-1])
        end for
        SubBytes(state)
        ShiftRows(state)
        AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1])
        out = state
End
```

ภาพประกอบ 2.28 ขั้นตอนวิธีเข้ารหัสแบบ AES (Mogollon, 2007)

จากภาพประกอบ 2.28 เป็นขั้นตอนวิธีการเข้ารหัสแบบ AES โดย AES แบ่งข้อมูลเป็นอาร์เรย์ 4 แถว แต่ละแถวมีจำนวนคอลัมน์ขึ้นอยู่กับขนาดความยาวของกุญแจ ถ้าขนาดของกุญแจเป็น 128 บิต แต่ละคอลัมน์ประกอบด้วยข้อมูล 32 บิต โดยเริ่มการทำงานรอบแรกด้วย AddRoundKey และแต่ละรอบประกอบด้วย SubBytes ShiftRows MixColumns และ AddRoundKey ซึ่งในรอบสุดท้ายประกอบด้วย SubBytes ShiftRows และ AddRoundKey โดยการทำงานในแต่ละรอบอธิบายได้ดังนี้ คือ

1) SubBytes เป็นการแทนข้อมูลแต่ละไบต์ด้วยค่าจากตารางค่าคงที่ค่าหนึ่ง โดยค่าที่ใช้เลือกจากค่าของข้อมูลเดิม

2) ShiftRows เป็นการเปลี่ยนแปลงตำแหน่งข้อมูล สำหรับการเข้ารหัสข้อมูลจะถูกเลื่อนไปทางซ้าย โดยที่ข้อมูลในแถวแรกจะไม่ถูกเลื่อนตำแหน่ง ส่วนการถอดรหัสจะเลื่อนตำแหน่งข้อมูลไปทางขวา โดยแถวแรกจะไม่ถูกเลื่อนตำแหน่งเหมือนกัน

3) MixColumns เป็นการนำข้อมูลในแต่ละคอลัมน์คูณกับค่าคงที่

4) AddRoundKey เป็นการนำข้อมูลมา XOR (Exclusive OR) กับกุญแจย่อยที่ถูกสร้างมาก่อนแล้ว

2) อัลกอริทึม RSA

RSA ย่อมาจากนามสกุลของนักวิจัย คือ Rivest Shamir และ Adleman (Mogollon, 2007) เป็นอัลกอริทึมเข้ารหัสแบบกุญแจอสมมาตร โดยการพัฒนาของ Ronald Rivest AdiShamir และ Leonard Adleman ในปี ค.ศ. 1978 โดยการเข้ารหัสจะได้กุญแจตัวหนึ่งในการเข้ารหัสและกุญแจอีกตัวหนึ่งในการถอดรหัส ซึ่งกุญแจสองตัวนี้มีความสัมพันธ์กันในทางคณิตศาสตร์ โดยขั้นตอนการทำงานของอัลกอริทึม RSA มีดังนี้

2.1) สุ่มเลือกจำนวนเฉพาะขนาดใหญ่ 2 จำนวน คือ P และ Q

2.2) คำนวณหาค่า N ด้วยสมการที่ (2.1)

$$N = P \times Q \tag{2.1}$$

2.3) คำนวณหาค่า $\varphi(n)$ ด้วยสมการที่ (2.2)

$$\varphi(n) = \varphi(P \times Q) = (P-1) \times (Q-1) \tag{2.2}$$

2.4) เลือกกุญแจสาธารณะ (E) เพื่อใช้เข้ารหัสข้อความ ซึ่ง E เป็นจำนวนเฉพาะสัมพัทธ์ $\varphi(n)$ ที่ทำให้ $\gcd(E, \varphi(n)) = 1$

2.5) คำนวณหาค่ากุญแจส่วนตัว (D) เพื่อใช้ในการถอดรหัสข้อความ โดยใช้ขั้นตอนวิธี Extended Euclid ดังสมการที่ (2.3) แล้วแทนที่กลับ (Modular Inverse) และทำให้สมการที่ (2.4) เป็นจริง

$$(E \times D) + (\varphi(n) \times Y) = 1 \tag{2.3}$$

$$(E \times D) \bmod \varphi(n) = 1 \tag{2.4}$$

2.6) เข้ารหัสข้อความ โดยคำนวณข้อความไซเฟอร์ (C) ได้จากข้อความต้นฉบับ (M) ด้วยสมการที่ (2.5)

$$C = M^E \bmod N \tag{2.5}$$

2.7) ส่งข้อความไซเฟอร์ที่ได้จากขั้นตอน 2.6 ไปยังผู้รับ

2.8) ผู้รับถอดรหัสข้อความไซเฟอร์(C) ได้เป็นข้อความต้นฉบับ(M) ด้วยสมการที่ (2.7)

$$M = C^D \bmod N \tag{2.7}$$

3) อัลกอริทึม SHA-1

SHA-1 หรือ Secure Hash Standards (Mogollon, 2007) เป็นการย่อยข้อความ ซึ่งถูกออกแบบมาเพื่อให้ไม่สามารถคำนวณหาข้อความที่สอดคล้องกับข้อความที่ย่อยแล้วได้ หรือไม่สามารถหาข้อความสองข้อความที่แตกต่างกันเมื่อย่อยแล้วได้ข้อความย่อยเดียวกันได้ ซึ่งอัลกอริทึมนี้ได้รับการพัฒนาโดย Information Processing Standards (FIPS) ในปี ค.ศ. 1995 ซึ่งใช้กับขนาดข้อความไม่เกิน 2^{64} บิต และเมื่อย่อยแล้วได้ข้อความขนาด 160 บิต

2.4 ความปลอดภัยของเว็บเซอร์วิส (Web Service Security)

การนำเว็บเซอร์วิสไปใช้งานในทางปฏิบัติจำเป็นต้องพิจารณาถึงเรื่องความปลอดภัย เนื่องจากเว็บเซอร์วิสพัฒนาด้วย XML จึงมีมาตรฐานความปลอดภัยของ XML ดังนี้

2.4.1 XML Signature

XML Signature (Mogollon, 2007) ถูกกำหนดโดย W3C เป็นการลงนามเพื่อรับรองเอกสาร ทำให้ข้อมูลมีความปลอดภัยในด้านการมีบูรณภาพ (Integrity) ของข้อความและการรับรองข้อความ โครงสร้างของ XML Signature แสดงดังภาพประกอบ 2.29 ซึ่งโครงสร้างของ XML Signature ประกอบด้วย 3 ส่วนสำคัญดังนี้

1) หมายเลข (1) คือส่วนของแท็ก <SignedInfo> เก็บข้อมูลเกี่ยวกับการลงลายเซ็น ประกอบด้วยแท็กย่อยภายในซึ่งอธิบายได้ดังนี้

บรรทัดที่ 3 คือแท็ก <CanonicalizationMethod/> ระบุอัลกอริทึมสำหรับ Canonicalize ก่อนที่จะผ่านฟังก์ชันแฮช

บรรทัดที่ 4 คือแท็ก <SignatureMethod/> ระบุอัลกอริทึมที่ใช้ในการลงลายเซ็นดิจิทัล เพื่อเก็บค่าในแท็ก <SignatureValue> ซึ่งข้อความจะถูกลงนามแบบเดียวกับที่ทำลายเซ็นดิจิทัล ข้อความจะผ่านฟังก์ชันแฮช และถูกลงนามด้วยการเข้ารหัสลับแบบสมมาตรหรือการเข้ารหัสลับแบบอสมมาตร เช่น SHA-1 และ RSA เป็นต้น

บรรทัดที่ 5 คือแท็ก <Reference> ประกอบด้วยแท็กย่อยภายในต่างๆ

บรรทัดที่ 6 คือแท็ก <Transforms> ระบุการแปลงข้อมูลก่อนฟังก์ชันแฮช เช่น Enveloped Enveloping หรือ Detached Signature เป็นต้น

บรรทัดที่ 7 คือแท็ก <DigestMethod> ระบุอัลกอริทึมในฟังก์ชันแฮช

บรรทัดที่ 8 คือแท็ก <DigestValue> ระบุข้อความย่อย

2) หมายเลข (2) คือส่วนของแท็ก <SignatureValue> ระบุค่าลายเซ็นดิจิทัล

3) หมายเลข (3) คือส่วนของแท็ก <KeyInfo> ระบุกุญแจที่ใช้สำหรับลงลายเซ็นดิจิทัล

```
1.  <Signature ID?>
2.      <SignedInfo>                                          (1)
3.          <CanonicalizationMethod/>
4.          <SignatureMethod/>
5.          <Reference URI? >
6.              (<Transforms>)?
7.              <DigestMethod>
8.              <DigestValue>
9.          </Reference>)+
10.     </SignedInfo>
11.     <SignatureValue>                                      (2)
12.     <KeyInfo>)?
13.         <KeyName>
14.         <KeyValue>
15.         <RetrievalMethod>                                 (3)
16.         <X509Data>
17.         <PGPData>
18.         <SPKIData>
19.     </KeyInfo>
20.     <Object ID?>)*
21. </Signature>
```

ภาพประกอบ 2.29 โครงสร้างของ XML Signature

XML Signature (XML Security, 2013: Online) แบ่งได้เป็น 3 ประเภทคือ

1) Enveloped Signature คือแท็ก <Signature> อยู่ภายในเอกสาร XML

2) Enveloping Signature คือแท็ก <Signature> เป็นแท็กแรก (Root Tag) และเอกสาร XML อยู่ภายในแท็ก <Signature>

3) Detached Signature คือแท็ก <Signature> และเอกสาร XML เป็นอิสระต่อกัน โดยแท็ก <Signature> อยู่นอกเอกสาร XML

1. <Root> 2. <Signature> 3. … 4. </Signature> 5. </Root>	1. < Signature> 2. <Root > 3. … 4. </Root> 5. </ Signature >	1. <Signature> 2. … 3. </Signature> 4. <Root > 5. … 6. </Root>
a) Enveloped Signature	b) Enveloping Signature	c) Detached Signature

ภาพประกอบ 2.30 Enveloped Enveloping และ Detached Signature

2.4.2 XML Encryption

XML Encryption (Mogollon, 2007) เป็นวิธีการเข้ารหัสมีการแทนที่ข้อมูลที่ต้องการเข้ารหัสด้วยแท็ก **<EncryptedData>** ซึ่งวิธีการนี้เป็นการรับประกันว่าข้อความ XML ที่ถูกเข้ารหัสแล้วจะเป็นความลับ (Confidentiality) โดยสามารถเลือกเข้ารหัสเอกสาร XML ได้ทั้งหมดหรือเลือกแค่บางส่วน ซึ่งโครงสร้างของ XML Encryption แสดงดังภาพประกอบ 2.31

```
1.  <EncryptedData Id? Type? MimeType? Encoding?>
2.      <EncryptionMethod/>?                                   (1)
3.      <ds:KeyInfo>
4.          <EncryptedKey>?
5.           <AgreementMethod>?
6.          <ds:KeyName>?                                      (2)
7.          <ds:RetrievalMethod>?
8.          <ds:*>?
9.      </ds:KeyInfo>?
10.     <CipherData>
11.         <CipherValue>?                                     (3)
12.         <CipherReference URI?>?
13.     </CipherData>
14.     <EncryptionProperties>?                                (4)
15. </EncryptedData>
```

ภาพประกอบ 2.31 โครงสร้างของ XML Encryption

จากภาพประกอบ 2.31 แท็ก <EncryptedData> แทนข้อมูลที่ถูกเข้ารหัสแล้ว ซึ่งภายในแท็กจะประกอบด้วย 4 ส่วนสำคัญ ดังนี้

1) หมายเลข (1) คือส่วนของแท็ก <EncryptionMethod> จัดเก็บวิธีการในการเข้ารหัสข้อความ

2) หมายเลข (2) คือส่วนของแท็ก <KeyInfo> จัดเก็บข้อมูลกุญแจในการเข้ารหัสและถอดรหัสข้อความ หากกุญแจถูกเข้ารหัสจะประกอบด้วยแท็กย่อยภายในแท็กนี้เพื่อจัดเก็บและอธิบายกุญแจที่ถูกเข้ารหัส

3) หมายเลข (3) คือส่วนของแท็ก <CipherData> จัดเก็บข้อความที่ถูกเข้ารหัสแล้ว

4) หมายเลข (4) คือส่วนของแท็ก <EncryptionProperties> จัดเก็บข้อมูลเพิ่มเติมสำหรับการเข้ารหัสข้อความ เช่น ข้อมูลเวลาและวันที่ เป็นต้น

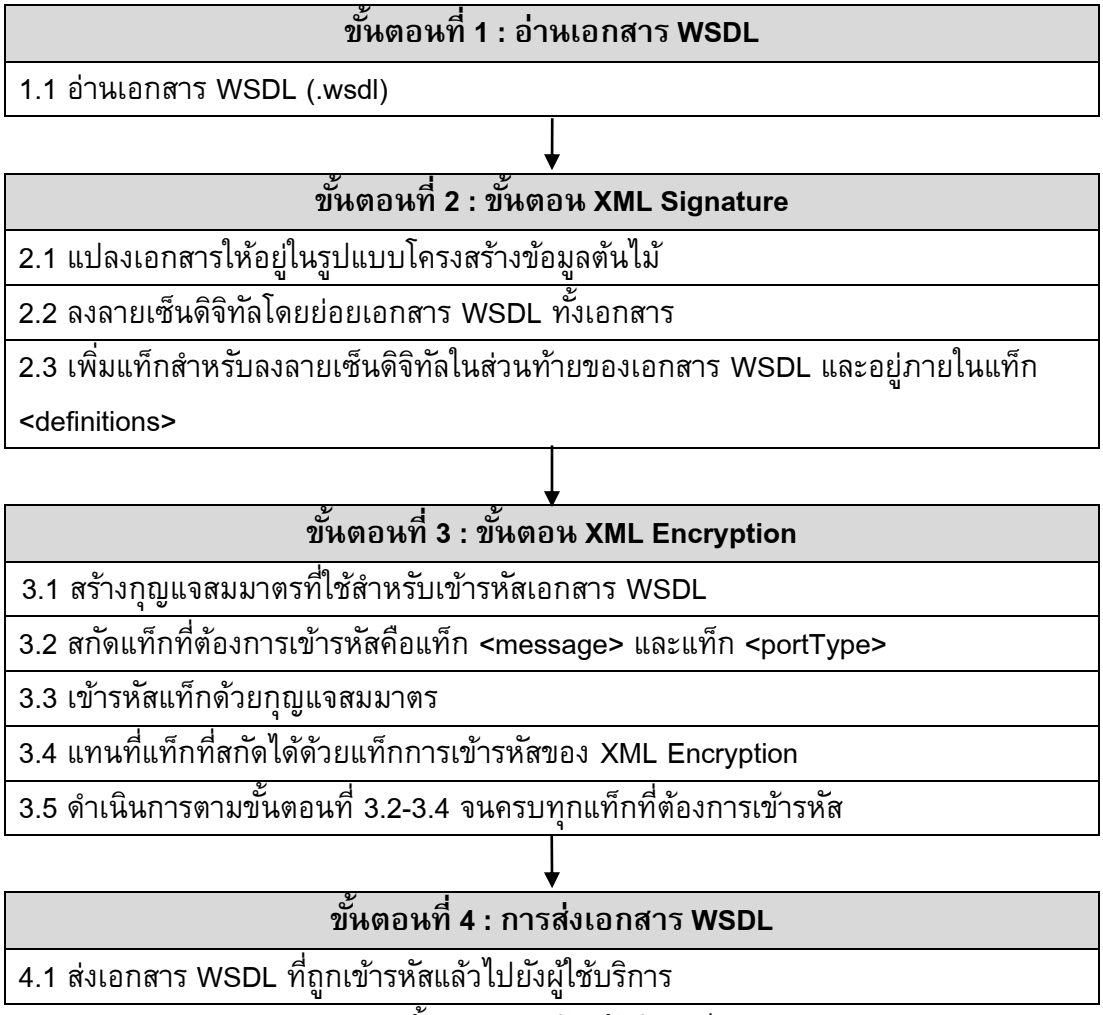
2.5 งานวิจัยที่เกี่ยวข้อง

เพื่อแก้ปัญหาภัยคุกคามของเอกสาร WSDL ในเว็บเซอร์วิส ได้มีการเสนอวิธีการป้องกันความปลอดภัยของเอกสาร WSDL หรือเรียกว่า “XMLSig_Enc” (Mirtalebi และ Khayyambashi, 2012) โดยใช้มาตรฐานความปลอดภัยของ XML คือ XML Signature และ XML Encryption และเข้ารหัสเอกสาร WSDL เพียงบางส่วน คือ แท็ก <message> และแท็ก <portType> ซึ่งประกอบด้วยกระบวนการเข้ารหัสและถอดรหัสข้อมูลในเอกสาร WSDL ดังต่อไปนี้

2.5.1 ขั้นตอนการเข้ารหัสด้วยวิธีการ XMLSig_Enc

ขั้นตอนการเข้ารหัสเอกสาร WSDL ด้วยวิธี XMLSig_Enc เริ่มต้นด้วยการอ่านเอกสาร WSDL ที่ต้องการเข้ารหัส หลังจากนั้นแปลงเอกสาร WSDL ให้อยู่ในรูปแบบโครงสร้างข้อมูลต้นไม้ การอ่านข้อมูลในเอกสาร WSDL จะใช้การอ่านแบบ DOM แล้วย่อยข้อความในเอกสาร WSDL ทั้งเอกสารเพื่อลงลายเซ็นดิจิทัลโดยใช้กุญแจส่วนตัวของผู้ให้บริการ ด้วยวิธีการเข้ารหัสแบบกุญแจอสมมาตร โดยแท็กที่ระบุการลงลายเซ็นดิจิทัลคือแท็ก <Signature> ซึ่งระบุแท็กนี้ในส่วนท้ายของเอกสาร WSDL และอยู่ภายในแท็ก <definitions> และหลังจากนั้นสร้างกุญแจสมมาตรและแลกเปลี่ยนกุญแจสมมาตรระหว่างผู้ให้บริการและผู้ใช้บริการ และสกัดแท็กที่สำคัญในเอกสาร WSDL คือแท็ก <message> และแท็ก <portType> เพื่อเข้ารหัสแท็กนี้ด้วยขั้นตอนวิธีการเข้ารหัสแบบกุญแจสมมาตร และแทนที่แท็กที่สกัดได้ด้วยแท็กการเข้ารหัสของ

XML Encryption คือ <EncryptedData> ทำเช่นนี้ไปจนครบจำนวนแท็กที่ต้องการเข้ารหัส เมื่อเข้ารหัสเอกสาร WSDL เสร็จสิ้นผู้ให้บริการส่งเอกสาร WSDL ที่ได้เข้ารหัสแล้วไปยังผู้ขอใช้บริการ อธิบายดังภาพประกอบ 2.32



ภาพประกอบ 2.32 ขั้นตอนการเข้ารหัสด้วยวิธีการ XMLSig_Enc

2.5.2 ขั้นตอนการถอดรหัสด้วยวิธีการ XMLSig_Enc

ขั้นตอนการถอดรหัสเอกสาร WSDL ด้วยวิธีการ XMLSig_Enc เริ่มต้นด้วยการอ่านเอกสาร WSDL ที่ต้องการถอดรหัส หลังจากนั้นแปลงเอกสาร WSDL ให้อยู่ในรูปแบบโครงสร้างข้อมูลต้นไม้ การอ่านข้อมูลในเอกสาร WSDL จะใช้การอ่านแบบ DOM แล้วย่อยข้อความในเอกสาร WSDL ทั้งเอกสารเพื่อตรวจสอบการลงลายเซ็นดิจิทัลโดยใช้กุญแจสาธารณะของผู้ให้บริการ ด้วยวิธีการเข้ารหัสแบบกุญแจอสมมาตร ถ้าย่อยข้อความแล้วได้ข้อความเหมือนกัน แสดงว่าเอกสาร WSDL ไม่ถูกปลอมแปลง หลังจากนั้นลบแท็กที่ระบุการลงลายเซ็นดิจิทัลคือแท็ก <Signature> ออกจากเอกสาร WSDL และสกัดแท็กที่ใช้ในการเข้ารหัส

คือแท็ก <EncryptedData> เพื่อถอดรหัสแท็กนี้ ด้วยขั้นตอนวิธีการเข้ารหัสแบบกุญแจสมมาตร โดยกุญแจสมมาตรมีการแลกเปลี่ยนกุญแจกันก่อนระหว่างผู้ให้บริการและผู้ใช้บริการ และแทนที่แท็กที่สกัดได้ด้วยแท็กที่ถูกถอดรหัสแล้วในเอกสาร WSDL เดิม ทำเช่นนี้ไปจนครบจำนวนแท็กที่ถูกเข้ารหัส เมื่อถอดรหัสเอกสาร WSDL เสร็จสิ้นผู้ใช้บริการนำเอกสาร WSDL ไปเรียกใช้งานเว็บเซอร์วิส อธิบายดังภาพประกอบ 2.33

ขั้นตอนที่ 1 : อ่านเอกสาร WSDL
1.1 อ่านเอกสาร WSDL ที่ถูกเข้ารหัสแล้ว (.wsdl)

↓

ขั้นตอนที่ 2 : Verify XML Signature
2.1 แปลงเอกสารให้อยู่ในรูปแบบโครงสร้างข้อมูลต้นไม้
2.2 ตรวจสอบความถูกต้องของการลงลายเซ็นดิจิทัล (Verify XML Signature)
2.3 ถ้าข้อมูลในเอกสารถูกต้องลบแท็กที่ใช้สำหรับลงลายเซ็นดิจิทัลจากเอกสาร WSDL

↓

ขั้นตอนที่ 3 : การถอดรหัส XML Encryption
3.1 สกัดแท็กที่ใช้สำหรับเข้ารหัสเอกสาร WSDL คือแท็ก <EncryptedData>
3.2 ถอดรหัสแท็กนี้ด้วยกุญแจสมมาตร
3.3 แทนที่ข้อมูลที่ถอดรหัสได้ในแท็กเดิม
3.4 ดำเนินการตามขั้นตอนที่ 3.1-3.3 จนครบทุกแท็กที่ต้องการถอดรหัส

↓

ขั้นตอนที่ 4 : การนำไปใช้งาน
4.1 ผู้ใช้บริการนำเอกสาร WSDL ไปเรียกใช้บริการเว็บเซอร์วิส

ภาพประกอบ 2.33 ขั้นตอนการถอดรหัสด้วยวิธีการ XMLSig_Enc

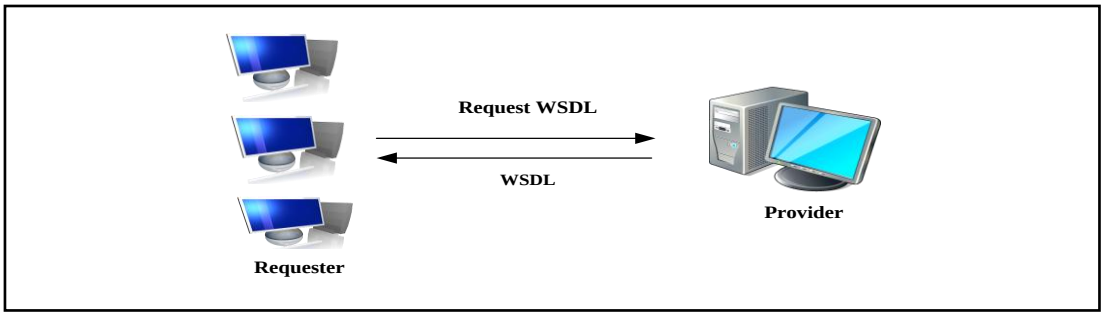
บทที่ 3

การวิเคราะห์และออกแบบกลไกสำหรับความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส

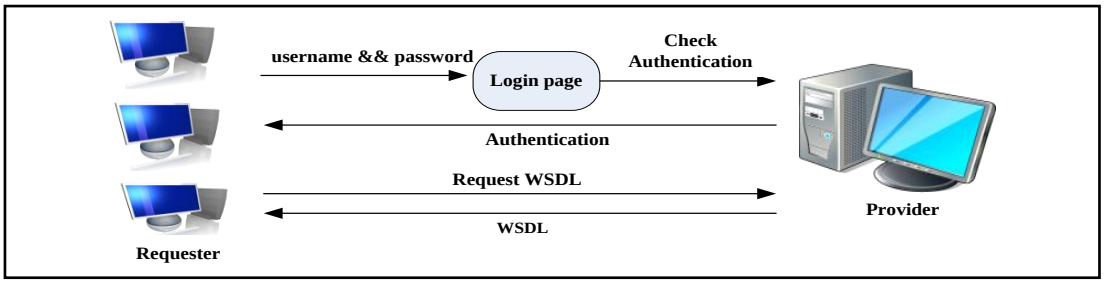
โดยปกติแล้วเอกสาร WSDL ไม่ได้ถูกทำให้เข้ารหัส ดังนั้นในการใช้งานจริงเอกสาร WSDL อาจถูกโจมตีโดยผู้ไม่ประสงค์ดีได้ ทำให้เกิดการขโมยข้อมูลในเว็บเซอร์วิสที่ให้บริการเซอร์วิสนั้น วิทยานิพนธ์นี้จึงวิเคราะห์และออกแบบกลไกสำหรับความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส เพื่อช่วยลดระยะเวลาในการเข้ารหัสและถอดรหัสเอกสาร WSDL อีกทั้งทำให้เอกสาร WSDL มีความปลอดภัยและถูกต้องสมบูรณ์เพื่อนำไปเรียกใช้งานเว็บเซอร์วิสต่อไป

3.1 วิเคราะห์การโจมตีเอกสาร WSDL

การใช้งานเว็บเซอร์วิสส่วนใหญ่ไม่ได้คำนึงถึงความปลอดภัยของเอกสาร WSDL โดยทั่วไปผู้ให้บริการเซอร์วิส เปิดให้เรียกใช้งานเซอร์วิสโดยประกาศเซอร์วิสที่ให้บริการหน้าเพจของผู้ให้บริการหรือลงทะเบียนเซอร์วิสที่ UDDI ผู้ใช้บริการสามารถเข้าถึงเอกสาร WSDL เพื่อนำเซอร์วิสนั้นไปใช้งานได้ โดยที่ไม่มีการปกปิดเอกสาร WSDL ตัวอย่างเช่น การให้บริการเว็บเซอร์วิสของกรมสรรพากร (กรมสรรพากร, 2012: Online) ซึ่งการร้องขอเอกสาร WSDL โดยทั่วไป มี 2 รูปแบบคือ การเข้าถึงเอกสาร WSDL ได้โดยตรงจากหน้าเพจของผู้ให้บริการ หรือต้องเป็นสมาชิกของผู้ให้บริการเว็บเซอร์วิสนั้นแสดงดังภาพประกอบ 3.1 (ก) และ 3.1 (ข) ตามลำดับ

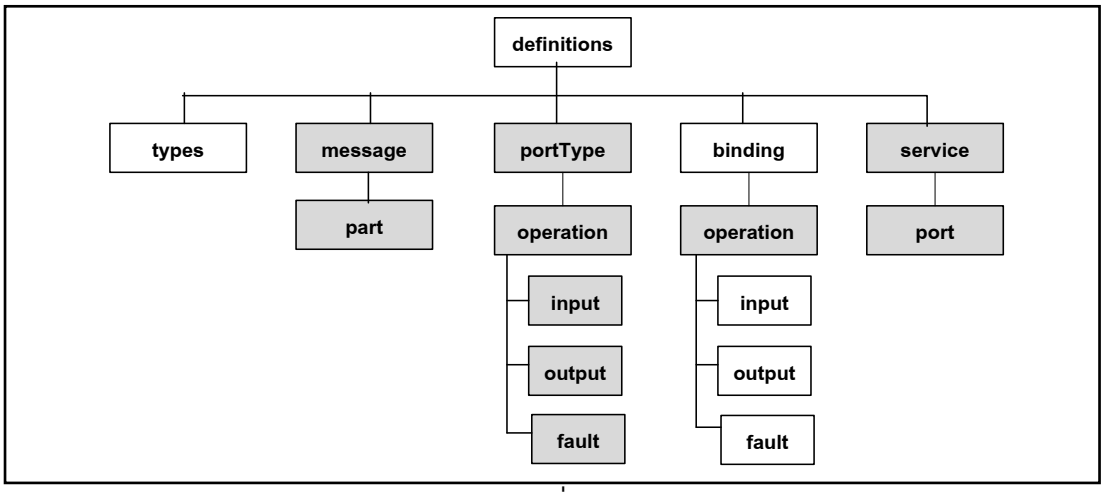


ภาพประกอบ 3.1 (ก) การร้องขอเอกสาร WSDL แบบเข้าถึงเอกสารโดยตรง



ภาพประกอบ 3.1 (ข) การร้องขอเอกสาร WSDL แบบเป็นสมาชิก

ดังนั้นผู้โจมตี (Hacker) นำเอกสาร WSDL มาวิเคราะห์ว่าเอกสาร WSDL ประกอบด้วยเซอร์วิสอะไรให้เรียกใช้งานบ้าง โดยวิเคราะห์ได้จากแท็ก <operation> ภายใต้แท็ก <portType> และแท็ก <binding> แท็ก <message> ระบุพารามิเตอร์ที่เรียกใช้งาน และแท็ก <service> ระบุ URL ที่เรียกใช้งานเซอร์วิสนั้นๆ ทำให้ผู้ไม่ประสงค์ดีโจมตีเซอร์วิสนั้นได้

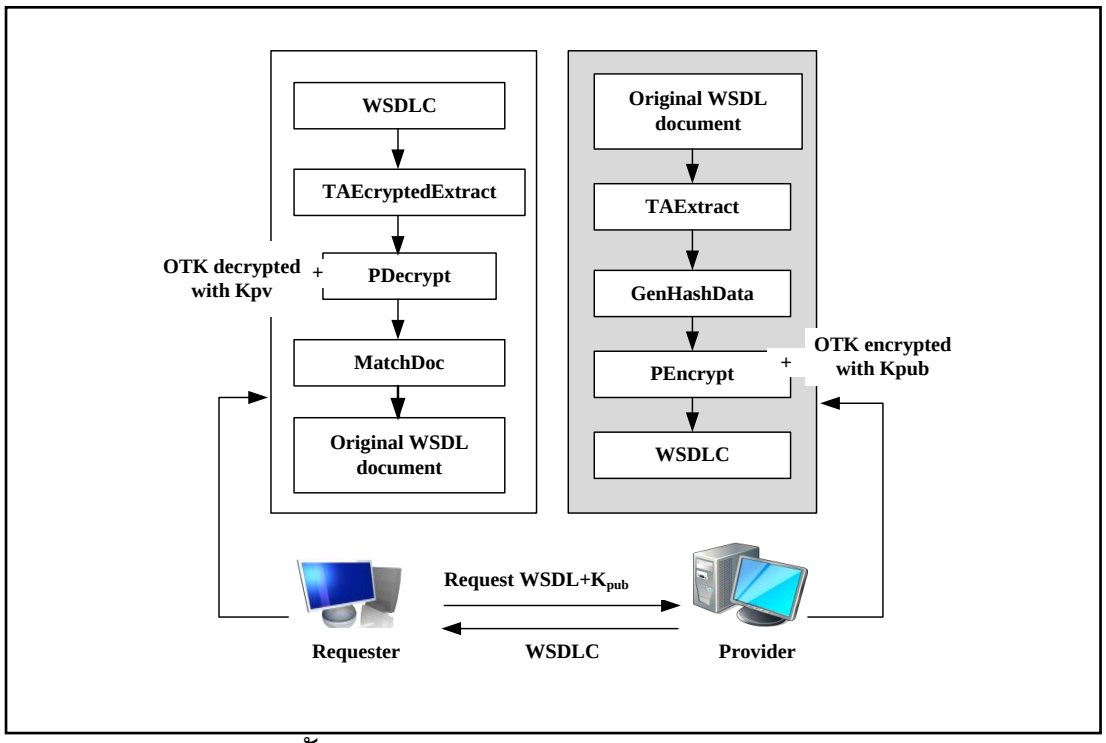


ภาพประกอบ 3.2 แท็กที่เข้ารหัสของ WSDL1.1

เพื่อทำให้เอกสาร WSDL มีความปลอดภัยสำหรับการนำไปใช้งาน ผู้วิจัยจึงเสนอกลไกการเข้ารหัสและถอดรหัสเอกสาร WSDL เพียงบางส่วน โดยเข้ารหัสและถอดรหัสแอททริบิวต์ในแท็กดังแสดงในภาพประกอบ 3.2 คือค่าแอททริบิวต์ในแท็กของ <message> <portType> <service> และแท็ก <operation> ภายใต้แท็ก <binding> โดยเข้ารหัสแท็ก <message> เพราะว่าแท็ก <message> เป็นแท็กที่ระบุพารามิเตอร์และชนิดของข้อมูลในการแลกเปลี่ยนระหว่างผู้ให้บริการและผู้ใช้บริการ และเข้ารหัสแท็ก <portType> และแท็ก <operation> ภายใต้แท็ก <binding> เพราะแท็กเหล่านี้ระบุเซอร์วิสสำหรับเรียกใช้งานไว้ภายในเอกสาร WSDL และเพื่อป้องกันการโจมตีแบบ WSDL Scanning ดังที่อธิบายในบทที่ 2 และเข้ารหัสแท็ก <service> เพื่อปกปิด URL ไม่ให้ผู้โจมตีเปลี่ยนแปลง URL ในแท็กนี้ได้และ

ป้องกันการโจมตีแบบ Parameter Tampering ดังอธิบายไว้ในบทที่ 2 เพื่อทำให้เอกสาร WSDL มีความปลอดภัยและไม่สามารถโจมตีได้ และการเข้ารหัสเอกสารเพียงบางส่วนเพื่อลดระยะเวลา นอกจากนี้ผู้ใช้บริการยังได้รับเอกสาร WSDL ที่เป็นความลับและถูกต้องสมบูรณ์

สถาปัตยกรรมกลไกสำหรับความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส (Mechanism for Securing WSDL Web Service: SWSDL) แสดงดังภาพประกอบ 3.3 ซึ่งแบ่งการทำงานเป็น 2 ส่วนคือ 1) ส่วนการเข้ารหัสเอกสาร WSDL (Encryption) สำหรับฝั่งผู้ให้บริการเว็บเซอร์วิส 2) ส่วนการถอดรหัสเอกสาร WSDL (Decryption) สำหรับผู้ใช้บริการเว็บเซอร์วิส



ภาพประกอบ 3.3 สถาปัตยกรรมกลไกสำหรับความปลอดภัยของเอกสาร WSDL (SWSDL)

3.2 ส่วนการเข้ารหัสเอกสาร WSDL (Encryption)

ในส่วนนี้ทำหน้าที่ในการเข้ารหัสเอกสาร WSDL สำหรับผู้ให้บริการ เมื่อผู้ใช้บริการ (Requester) ร้องขอเอกสาร WSDL และส่งกุญแจสาธารณะ (K_{pub}) ของตนเองกับผู้ให้บริการ (Provider) การทำงานในส่วนนี้ประกอบไปด้วย การสกัดแท็กและค่าแอททริบิวต์ในเอกสาร WSDL (TAExtract) การกำหนดหมายเลขและกระบวนการแฮช (GenHashData) และการเข้ารหัสเอกสาร WSDL (PEncrypt) ซึ่งมีรายละเอียดดังต่อไปนี้

3.2.1 การสกัดแท็กและค่าแอททริบิวต์ในเอกสาร WSDL (TAExtract)

ขั้นตอนนี้เป็นขั้นตอนแรกของการเข้ารหัสเอกสาร WSDL ซึ่งอธิบายการทำงานได้ดังต่อไปนี้

1) ผู้ให้บริการนำเอกสาร WSDL มาเข้ารหัสตามคำขอเรียกใช้เซอร์วิสนั้นจากผู้ใช้บริการ

2) สกัดแท็กที่สำคัญภายในเอกสาร WSDL คือ แท็ก <message> <portType> <service> โดยภายในแท็กเหล่านี้จะมีแท็กย่อยต่างๆ และแท็ก <operation> ภายใต้แท็ก <binding>

3) แต่ละแท็กที่สกัดได้จะนำมาสกัดค่าแอททริบิวต์ในแต่ละแท็ก เพื่อนำไปใช้ในขั้นตอนต่อไป

ขั้นตอนวิธีในการสกัดแท็กและค่าแอททริบิวต์แสดงดังภาพประกอบ 3.4

```
1. Method Extract (WSDL)
2.    Extract  each <message>, <portType>, <service>  including their sub tags and
      <operation> in <binding> (TagName)
3.    For each TagName
4.          Extract Attribute value in TagName (Attr)
5.    EndFor
6. EndMethod
```

ภาพประกอบ 3.4 ขั้นตอนวิธีการสกัดแท็กและค่าแอททริบิวต์

3.2.2 การกำหนดหมายเลขและกระบวนการแฮช (GenHashData)

ขั้นตอนวิธีนี้ประกอบด้วยขั้นตอนย่อยภายใน 2 ขั้นตอน คือ

1) การกำหนดหมายเลข

ขั้นตอนนี้เป็นการกำหนดหมายเลขให้แต่ละค่าแอททริบิวต์ เนื่องจากแต่ละค่าแอททริบิวต์ที่สกัดได้จากขั้นตอนที่ 3.2.1 (TAExtract) ในเอกสาร WSDL อาจมีค่าแอททริบิวต์ที่เหมือนกัน เมื่อผ่านกระบวนการแฮชเพื่อย่อยข้อความจะได้ข้อความย่อยเหมือนกันด้วย ดังนั้นเพื่อป้องกันไม่ให้เกิดข้อความย่อยเดียวกันจึงต้องกำหนดหมายเลขให้แต่ละค่าแอททริบิวต์ ซึ่งค่าแอททริบิวต์ที่สกัดได้จะถูกแปลงให้อยู่ในรูปของโครงสร้างต้นไม้ของเอกสาร XML (XML Tree) โดยใช้ XML Parser ประเภท Document Object Model (DOM) (Benz และ Durant, 2003) เนื่องจากเมื่อสกัดค่าแอททริบิวต์ที่ต้องการได้แล้วจะนำค่าแอททริวบิวต์มาผ่าน

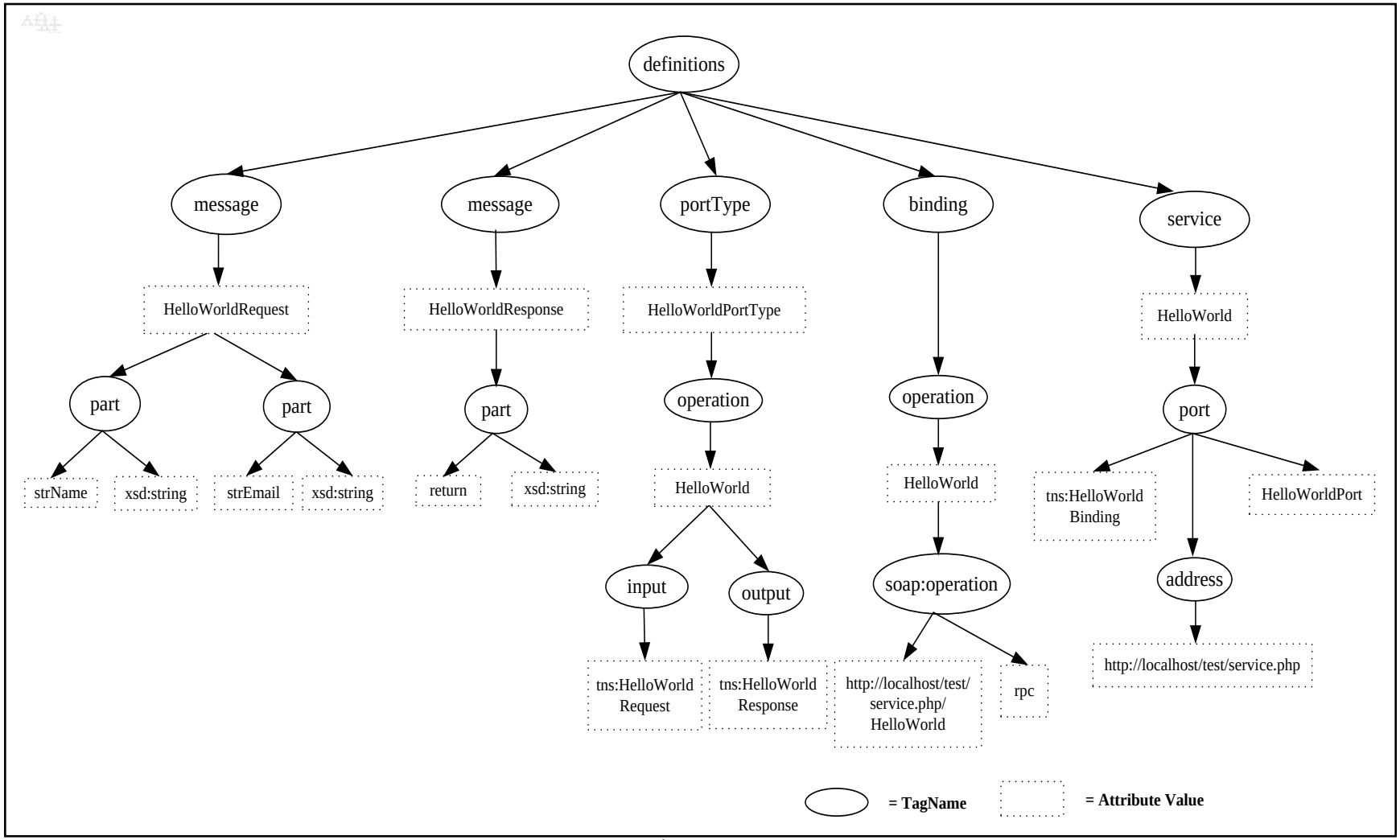
กระบวนการแฮชเพื่อให้ได้ค่าแฮชของแต่ละค่าแอททริบิวต์นั้นและแทนที่กลับเข้าไปในตำแหน่งค่าแอททริบิวต์เดิมที่สกัดมาได้ จึงใช้ XML Parser ประเภท DOM ซึ่งสามารถที่จะแก้ไขเนื้อหาในเอกสาร XML ได้ดังอธิบายไว้ในบทที่ 2 และการกำหนดหมายเลขเริ่มต้นจากโหนดแรกที่สกัดได้ (<message>, <portType>, <service> และ <binding>) เริ่มจากหมายเลข 0 และเพิ่มค่าหมายเลขไปเรื่อยๆจากซ้ายไปขวา หากมีโหนดลูกจะเพิ่มหมายเลขโดยคั่นด้วยเครื่องหมาย “.” เพื่อบอกลำดับขั้นและเพิ่มค่าหมายเลขไปเรื่อยๆจากซ้ายไปขวา หากในโหนดลูกมีค่าแอททริบิวต์มากกว่า 1 ค่าจะเพิ่มค่าหมายเลขอีกหมายเลขต่อท้ายค่าหมายเลขเดิมจากซ้ายไปขวา ทำให้แต่ละค่าแอททริบิวต์มีหมายเลขไม่ซ้ำกัน และเมื่อผ่านกระบวนการแฮชจะได้ผลลัพธ์เป็นข้อความย่อยต่างกัน ขั้นตอนวิธีการกำหนดหมายเลขและกระบวนการแฮชแสดงดังภาพประกอบ 3.5 ตัวอย่างแท็กที่ถูกสกัดของเอกสาร WSDL1.1 แสดงดังภาพประกอบ 3.6 รูปแบบโครงสร้างต้นไม้เมื่อสกัดแท็กและค่าแอททริบิวต์ของเอกสาร WSDL1.1 และรูปแบบโครงสร้างต้นไม้เมื่อกำหนดหมายเลขแอททริบิวต์ของเอกสาร WSDL1.1 แสดงดังภาพประกอบ 3.7 และ 3.8 ตามลำดับ

```
//Attr is Attribute Value
1.  Method GenHashData (Attr,TreeMsgNo)
2.   Set “ ” to TreeMsgNo
3.       For each Attr
4.              Assign number in each Attr (Attr_num)
5.              Hash_Value = H(Attr_num)
6.              Replace Attr with Hash_Value
7.              Concatenate Attr_num with  “|” to TreeMsgNo
8.       EndFor
9.  EndMethod
```

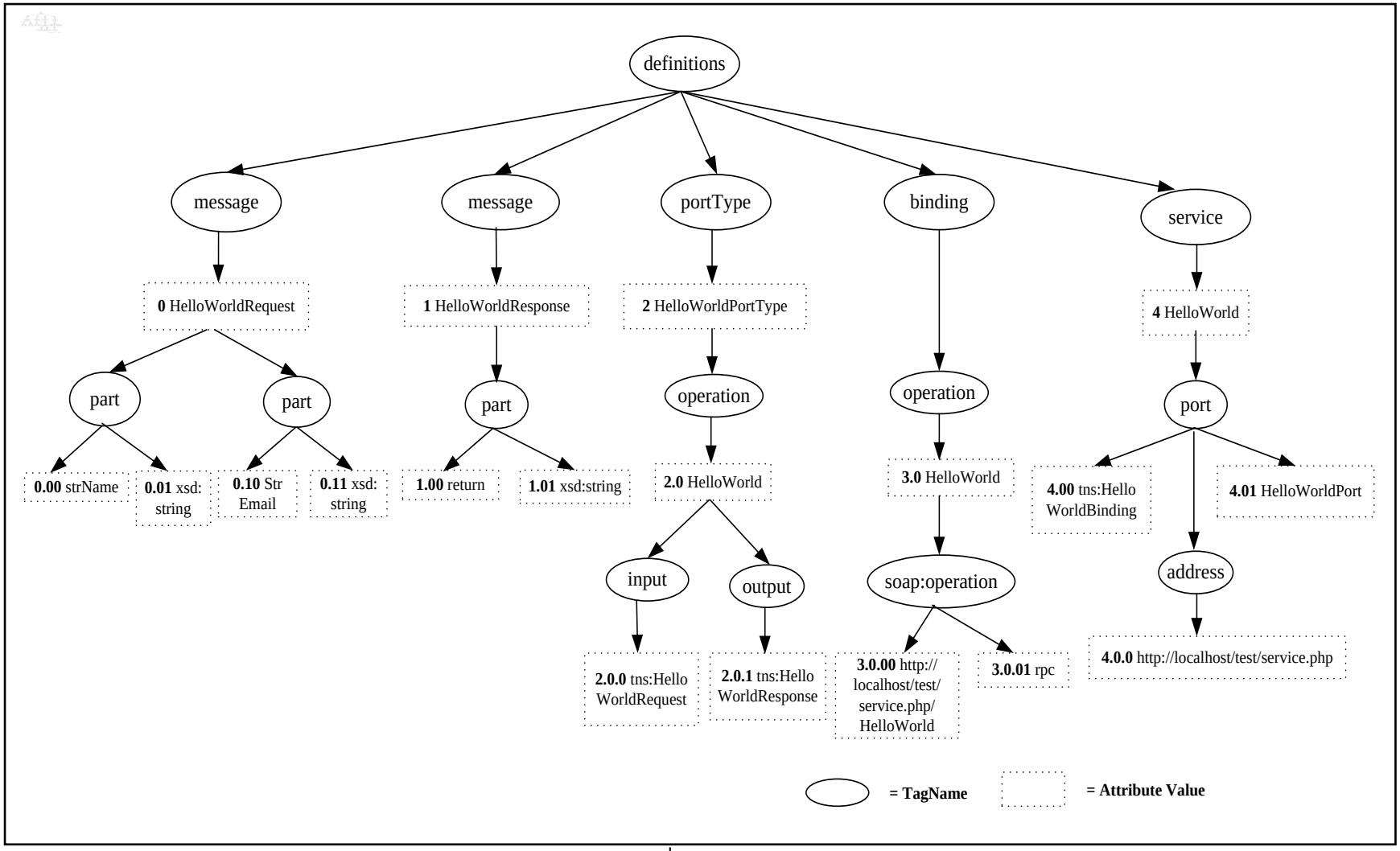
ภาพประกอบ 3.5 ขั้นตอนวิธีการกำหนดหมายเลขและกระบวนการแฮช

```
1.  <definitions targetNamespace="http://localhost/soap/HelloWorld">
2.    <types>
3.        <xsd:schema targetNamespace="http://localhost/soap/HelloWorld">
4.              <xsd:import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
5.              <xsd:import namespace="http://schemas.xmlsoap.org/wsdl/"/>
6.        </xsd:schema>
7.    </types>
8.    <message name="HelloWorldRequest">
9.              <part name="strName" type="xsd:string"/>
10.             <part name="strEmail" type="xsd:string"/>
11.   </message>
12.   <message name="HelloWorldResponse">
13.             <part name="return" type="xsd:string"/>
14.   </message>
15.   <portType name="HelloWorldPortType">
16.             <operation name="HelloWorld">
17.                       <input message="tns:HelloWorldRequest"/>
18.                       <output message="tns:HelloWorldResponse"/>
19.             </operation>
20.   </portType>
21.   <binding name="HelloWorldBinding" type="tns:HelloWorldPortType">
22.       <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
23.       <operation name="HelloWorld">
24.           <soap:operation soapAction="http://localhost/test/service.php/HelloWorld" style="rpc"/>
25.             <input>
26.             <soap:body use="encoded" namespace=""
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
27.             </input>
28.             <output>
29.             <soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
30.             </output>
31.       </operation>
32.   </binding>
33.   <service name="HelloWorld">
34.     <port name="HelloWorldPort" binding="tns:HelloWorldBinding">
35.             <soap:address location="http://localhost/test/service.php"/>
36.     </port>
37.   </service>
38. </definitions>
```

ภาพประกอบ 3.6 ตัวอย่างแท็กที่ถูกสกัดของเอกสาร WSDL1.1



ภาพประกอบ 3.7 รูปแบบโครงสร้างต้นไม้เมื่อสกัดแท็กและค่าแอททริบิวต์ของเอกสาร WSDL1.1



ภาพประกอบ 3.8 รูปแบบโครงสร้างต้นไม้เมื่อกำหนดหมายเลขแอททริบิวต์ของเอกสาร WSDL1.1

2) กระบวนการแฮช

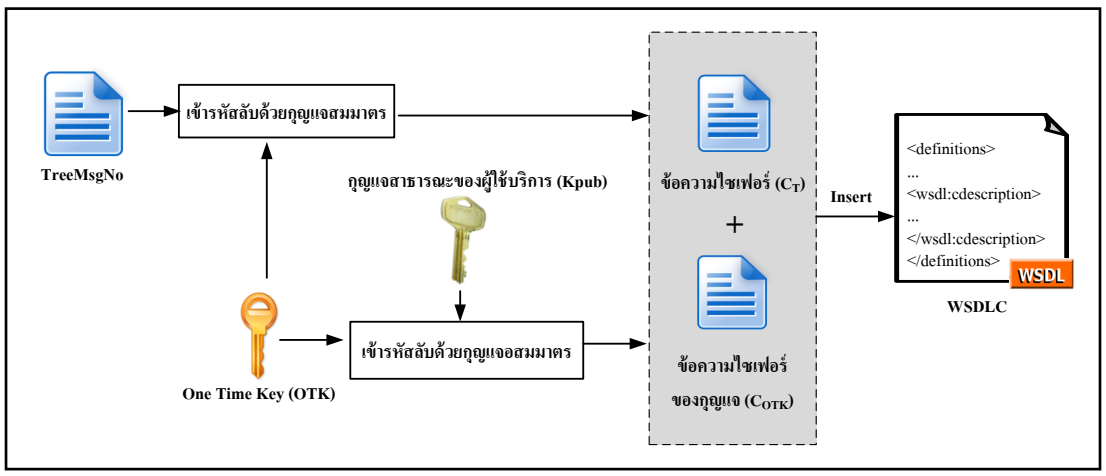
ขั้นตอนนี้เป็นการนำแต่ละค่าแอททริบิวต์ที่ผ่านขั้นตอนการกำหนดหมายเลขแล้วมาผ่านกระบวนการแฮชเพื่อย่อยข้อความ โดยมีวัตถุประสงค์เพื่อตรวจสอบความถูกต้องของข้อความ ข้อดีของกระบวนการแฮชคือผู้โจมตีไม่สามารถถอดรหัสข้อความย่อยได้ เพราะแฮชเป็นการทำงานเพียงทางเดียวดังอธิบายไว้ในบทที่ 2 ฟังก์ชันแฮชที่นำมาใช้ คือ SHA-1 จะได้ผลลัพธ์เป็นข้อความย่อยของแต่ละค่าแอททริบิวต์ ซึ่งข้อความย่อยที่ได้จะไม่ซ้ำกัน และนำข้อความย่อยที่ได้ไปแทนที่ค่าแอททริบิวต์เดิมในเอกสาร WSDL ดังภาพประกอบที่ 3.14 (ก) และนำค่าแอททริบิวต์ที่กำหนดหมายเลขแล้วรวมเป็นข้อความเดียวกันโดยแต่ละค่าแอททริบิวต์คั่นด้วยเครื่องหมาย “|” (TreeMsgNo) สำหรับนำไปใช้ในขั้นตอนการเข้ารหัส ดังภาพประกอบ 3.9

```
0 HelloWorldRequest|0.00 strName|0.01 xsd:string|0.10 strEmail|0.11 xsd:string|1
HelloWorldResponse|1.00 return|1.01 xsd:string|2 HelloWorldPortType|2.0 HelloWorld|2.0.0
tns:HelloWorldRequest|2.0.1 tns:HelloWorldResponse|3.0 HelloWorld|3.0.00
http://localhost/test/service.php/HelloWorld|3.0.01 rpc|4 HelloWorld|4.00
tns:HelloWorldBinding|4.01 HelloWorldPort|4.0.0 http://localhost/test/service.php
```

ภาพประกอบ 3.9 การรวมแต่ละค่าแอททริบิวต์ (TreeMsgNo)

3.2.3 การเข้ารหัสเอกสาร WSDL (PEncrypt)

เป็นกระบวนการเข้ารหัสเอกสาร WSDL เพื่อให้เอกสารเป็นความลับ ปลอดภัยจากผู้ไม่ประสงค์ดี (ดูภาพประกอบ 3.10) โดยมีขั้นตอนวิธีดังภาพประกอบ 3.11 ซึ่งอธิบายได้ดังนี้



ภาพประกอบ 3.10 กระบวนการเข้ารหัสข้อมูลเอกสาร WSDL

```
1.  Method PEncrypt (TreeMsgNo,K_pub,WSDL)
2.      Generate One Time Key (OTK)
3.      C_T = E_OTK(TreeMsgNo)          //Encrypt TreeMsgNo with OTK
4.      C_OTK = E_K_pub(OTK)            //Encrypt OTK with K_pub
5.      Create <wsdl:cdescription> tag  in WSDL
6.      Insert algorithm in <wsdl:algorithm> tag
7.      Insert cipher key (C_OTK) in <wsdl:key> tag
8.      Insert cipher TreeMsgNo (C_T) in <wsdl:cipher> tag
9.   EndMethod
```

ภาพประกอบ 3.11 ขั้นตอนวิธีการเข้ารหัสเอกสาร WSDL

1) การเข้ารหัสเอกสาร WSDL

เป็นการนำข้อความที่ได้จากการรวมแต่ละค่าแอททริบิวต์ที่กำหนดหมายเลขแล้ว (TreeMsgNo) มาเข้ารหัสเอกสารเพียงครั้งเดียวโดยใช้กุญแจสมมาตร (One Time Key: OTK) ด้วยขั้นตอนวิธี AES ซึ่งกลไก SWSDL สร้างกุญแจสมมาตร OTK ขนาด 128 บิต ได้ข้อความไซเฟอร์ (C_T) และเข้ารหัสกุญแจสมมาตร OTK โดยใช้กุญแจสาธารณะของผู้ใช้บริการ (K_{pub}) ด้วยขั้นตอนวิธี RSA ได้ข้อความไซเฟอร์ของกุญแจ (C_{OTK}) โดยกำหนดให้เอกสาร WSDL ระบุส่วนที่เข้ารหัสไว้ภายในเอกสาร WSDL ในแท็ก <wsdl:cdescription>

2) การสร้างแท็ก <wsdl:cdescription>

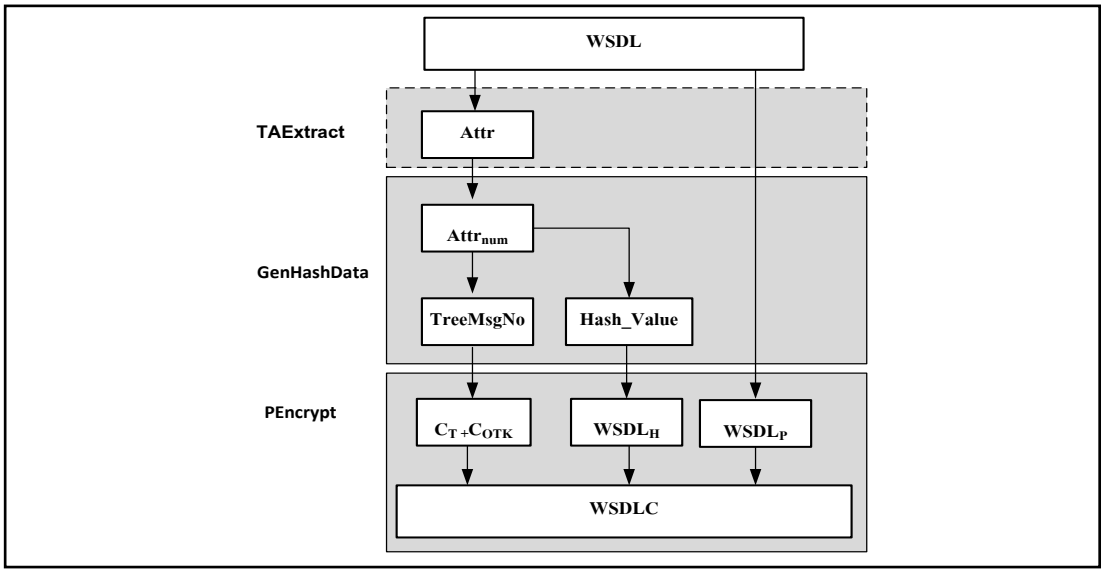
เพื่อให้เอกสาร WSDL เผยแพร่ข้อมูลที่มีความปลอดภัยได้ วิทยานิพนธ์นี้จึงสร้างแท็ก <wsdl:cdescription> เพิ่มเติมภายในเอกสาร WSDL เพื่อเก็บรายละเอียดเกี่ยวกับการเข้ารหัส โดยมีรูปแบบดังภาพประกอบ 3.12 และคำอธิบายรายละเอียดดังตาราง 3.1

```
1.  <wsdl:cdescription>
2.      <wsdl:algorithm> Encryption  Algorithm</wsdl:algorithm>
3.      <wsdl:key>Cipher Key</wsdl:key>
4.      <wsdl:cipher>Cipher Message </wsdl:cipher>
5.  </wsdl:cdescription>
```

ภาพประกอบ 3.12 รูปแบบแท็ก <wsdl:cdescription> ของเอกสาร WSDL

ตารางที่ 3.1 คำอธิบายแท็กต่างๆ ของแท็ก <wsdl:cdescription >

แท็ก	คำอธิบาย
<wsdl:cdescription>	ข้อมูลที่จำเป็นเกี่ยวกับการเข้ารหัสเอกสาร WSDL
<wsdl:algorithm>	อัลกอริทึมที่ใช้ในการเข้ารหัสเอกสาร WSDL
<wsdl:key>	ข้อความไซเฟอร์ของกุญแจที่ใช้ในการเข้าและถอดรหัสข้อมูลในเอกสาร WSDL
<wsdl:cipher>	ข้อความไซเฟอร์ของเอกสาร WSDL



ภาพประกอบ 3.13 ผลลัพธ์แต่ละขั้นตอนของการเข้ารหัสเอกสาร WSDL1.1

ภาพประกอบ 3.13 แสดงผลลัพธ์แต่ละขั้นตอนการเข้ารหัสเอกสาร WSDL1.1 ซึ่งอธิบายได้ดังนี้

ผลลัพธ์ของขั้นตอนของการสกัดแท็กเอกสาร WSDL (TAExtract) จะได้ค่าแอททริบิวต์ (Attr)

ผลลัพธ์ของขั้นตอนการกำหนดหมายเลขและกระบวนการแฮช (GenHashData) คือ การนำ Attr มากำหนดหมายเลขแล้วจะได้ค่าแอททริบิวต์ที่มีหมายเลขกำหนด ($Attr_{num}$) แล้วนำ $Attr_{num}$ มาต่อเป็นข้อความเดียวกันได้เป็นข้อความ (TreeMsgNo) และนำค่า $Attr_{num}$ มาผ่านกระบวนการแฮชจะได้ข้อความย่อย (Hash_Value)

ผลลัพธ์ของขั้นตอนการเข้ารหัสเอกสาร WSDL (PEncypt) คือ WSDL ที่เข้ารหัสแล้ว (WSDLC) ภายในเอกสารประกอบด้วย 1) เอกสาร WSDL ต้นฉบับส่วนที่ไม่ได้เข้ารหัส ($WSDL_P$) 2) เอกสาร WSDL ส่วนที่ผ่านกระบวนการแฮช ($WSDL_H$) 3) ข้อความไซ-เฟอร์ (C_T) และ 4) ข้อความไซเฟอร์ของกุญแจที่ใช้ในการเข้ารหัส (C_{OTK}) โดยตัวอย่างเอกสาร WSDL ที่เข้ารหัสแล้วแสดงดังภาพประกอบ 3.14 (ก) และ 3.14 (ข) ตามลำดับ

```
1.  <definitions targetNamespace="http://localhost/soap/HelloWorld">
2.    <types>
3.       <xsd:schema targetNamespace="http://localhost/soap/HelloWorld">
4.            <xsd:import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
5.            <xsd:import namespace="http://schemas.xmlsoap.org/wsdl/"/>
6.       </xsd:schema>
7.    </types>
8.    <message name="10A488FFA001363427F0347AF9A69874386366BE">
9.            <part name="4763A4017C2938E5EB149536387DCF059A352F36"
              type="EE476447480D921A62A88DF4A6EA4A9B19674882"/>
10.           <part name="AFEF9BF391FBD1A8190057AA5550676F35EF3648"
              type="1035544FDAF3FE2DD348D65DEB23AB7DAAE122D3"/>
11.   </message>
12.   <message name="DFA1ECF7CD03724AC5A87B43B91D5B275DA90B81">
13.           <part name="1AEBAEEA1D0A07079091141F5FFF6D5FFD2E21A6"
              type="46217B4AAC48EDA467D9C4D28DE35ED4EF48CEE6"/>
14.   </message>
15.   <portType name="58BEB2A543EEF9378C26AC4DB70E454B0FB20EBF">
16.           <operation name="1097EBADC9A6602BEF2A99D809A45FFAA01C9669">
17.              <input message="31DD02E7B9F4C5CD5B179641AAE255171A294B55"/>
18.             <output message="AA3A12F4A883878FDA9285BA546671D6E06B32E4"/>
19.           </operation>
20.   </portType>
21.  <binding name="HelloWorldBinding" type="tns:HelloWorldPortType">
22.       <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
23.     <operation name="31DEBA136530931B032ABB7405FDF49AB0ACCEC6">
24.      <soap:operation soapAction="2B33E44D1C3BA3FEC708655DB8B0948059DE361F"
           style="2DFD48B78F637DEBF12314654D025838BFD2BE0D"/>
25.           <input><soap:body use="encoded" namespace=""
              encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/></input>
26.           <output>
27.           <soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
28.           </output>
29.      </operation>
30.   </binding>
31.   <service name="71A1A47097015E6CA79F01187B88BF51DD0D254F">
32.      <port binding="DCF3C6DF3E5577AC80CF92392D79E457408C240E"
         name="3F67B076298A394BDAC78DA48F49B07676EC352F">
33.           <soap:address location="BE7B6281B55CC379548C5095872D86DF25FF2F4F"/>
34.      </port>
35.   </service>
```

ภาพประกอบ 3.14 (ก) ส่วนการแฮชของเอกสาร WSDL ที่เข้ารหัสแล้ว (WSDLC)

```
36.  <wsdl:cdescription>
37.  <wsdl:algorithm>AES/128/PKCS5Padding</wsdl:algorithm>
38.  <wsdl:key>7A1DD404E5890569769554503187CD820B8FFF8B3A6ED30E9F0D2B5E436AA87D41
                895CC771476EC8250B96134769EF2CC984D34395160B462EF8E3433453D3FD
39.  </wsdl:key>
40.  <wsdl:cipher>548CB1B927EE6947F294F180B16145D171AF9999E16CAFF53D03F101688BAD14
     1FF0F5104D0B0CE2315262D7CE97AB40EB607989A82D3B9B44B11A95EE49F3AC1B01F10AA4
     FF3C935122A8E6F8B70780A8834853B3FE82D07CCF68CE92BF758EC78630E9ED636F5D6B59
     263A52CB5B3FE3B1CADE7D146603CA89B22615A0645CB14E85AF8E3828165E8014AA2B699A
     36CF8A9E630B203EB4BB1AD1E245AF514F715ED46BA2AF0E3E725E4A792D2DB5A886740904
     70FF209E14D12BFA3E038485BD0357BB3D767B41279365557F33785FB3C333EBF2F66D06BF1
     7DE4564C15B64F0F181BBFD1C4B280A0CFC58FE7108D4AC8CA8A7F96C78AC775FF3A57E36
     F443D636082DA50C31519EFF903E6A74A147FC183482B057EB427C3B19A35A0EA5F4C495B9
     3C6BECF04B6829BD0C07769C11D7C99EDC17C85269B20FD20FF81F90E7E95D4FD98DE3CE
     4215F263731B7181D225DF68DF460857F361A6A10F928053855CD031C9A07D3E4B589BAFA00
     954112CED31F300D67E7AE25BCB74C9168E2FB75331E41DAC969261DFE763D8BA36AC1A4B
     42B22002937C4B4EDA1DAEF85F3C1A62FA8731D8C655EDEA01C57A8DE689BE
41.  </wsdl:cipher>
42.  </wsdl:cdescription>
43.  </definitions>
```

ภาพประกอบ 3.14 (ข) ส่วนเข้ารหัสของเอกสาร WSDL ที่เข้ารหัสแล้ว (WSDLC)

3.3 ส่วนการถอดรหัสเอกสาร WSDL (Decryption)

ในส่วนนี้ทำหน้าที่ในการถอดรหัสเอกสาร WSDL สำหรับผู้ใช้บริการ เมื่อผู้ใช้บริการได้รับเอกสาร WSDL ที่เข้ารหัสแล้วและแนบกุญแจในการเข้ารหัสไว้ภายในเอกสาร WSDL เพื่อนำมาถอดรหัส การทำงานในส่วนนี้ประกอบไปด้วย การสกัดแท็กส่วนเข้ารหัสเอกสาร WSDL (TAEcryptedExtract) การถอดรหัสเอกสาร WSDL (PDecrypt) และการเปรียบเทียบแอททริบิวต์ในเอกสาร WSDL (MatchDoc) ซึ่งมีรายละเอียดดังต่อไปนี้

3.3.1 การสกัดแท็กส่วนเข้ารหัสเอกสาร WSDL (TAEcryptedExtract)

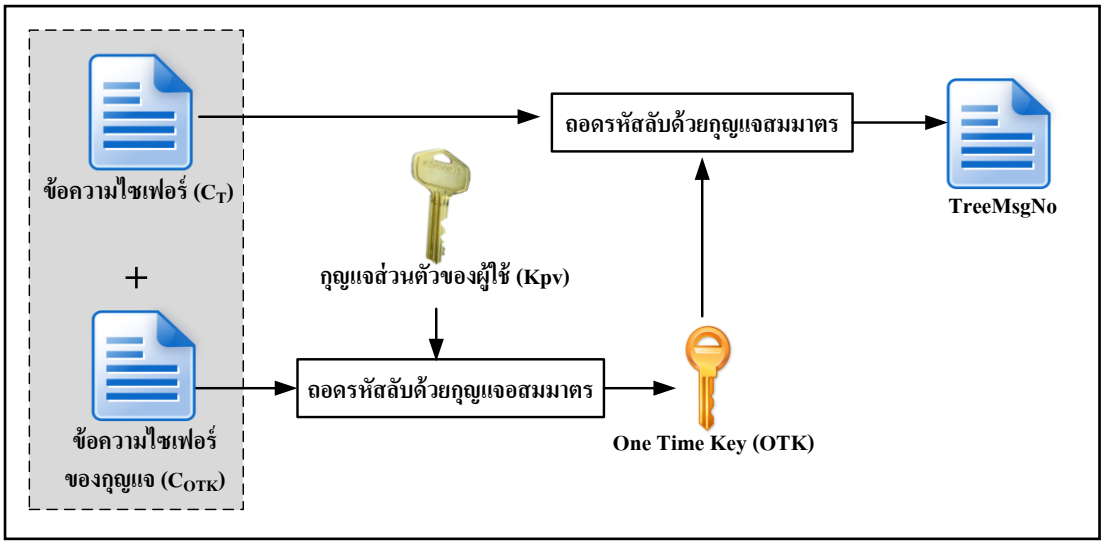
ขั้นตอนนี้เป็นขั้นตอนแรกในการถอดรหัสเอกสาร WSDL โดยผู้ใช้บริการนำเอกสาร WSDL ที่ต้องการถอดรหัส (WSDLC) สกัดแท็กในส่วนที่ใช้ในการเข้ารหัสเอกสาร WSDL คือ แท็ก <wsdl:cdescription> สกัดแท็กย่อยภายในแท็กนี้คือแท็ก <wsdl:key> และสกัดเนื้อหาภายในแท็กนี้จะได้ข้อความไซเฟอร์ของกุญแจ (C_{OTK}) และสกัดแท็ก <wsdl:cipher>

อีกทั้งสกัดเนื้อหาภายในแท็กนี้จะได้ข้อความไซเฟอร์ (C_T) ขั้นตอนวิธีในการสกัดแท็กส่วนของการเข้ารหัสเอกสาร WSDL แสดงดังภาพประกอบ 3.15

```
1.    Method  TAEcryptedExtract (WSDLC)
2.        Extract  <wsdl:cdescription>
3.        Extract  <wsdl:key> and Content
4.        Extract  <wsdl:cipher> and Content
5.    EndMethod
```

ภาพประกอบ 3.15 ขั้นตอนวิธีการสกัดแท็กส่วนเข้ารหัสเอกสาร WSDL

3.3.2 การถอดรหัสเอกสาร WSDL (PDecrypt)



ภาพประกอบ 3.16 กระบวนการถอดรหัสเอกสาร WSDL

ภาพประกอบ 3.16 กระบวนการถอดรหัสเอกสาร WSDL จะทำงานที่ฝั่งเครื่องผู้ใช้บริการ โดยมีขั้นตอนการทำงาน ดังนี้

1) ถอดรหัสข้อความไซเฟอร์ของกุญแจ (C_{OTK}) โดยใช้กุญแจส่วนตัวของผู้ใช้บริการ (K_{pv}) ด้วยขั้นตอนวิธี RSA จะได้กุญแจสมมาตร One Time Key (OTK)

2) นำกุญแจสมมาตร One Time Key ที่ได้ไปถอดรหัสข้อความไซเฟอร์ (C_T) ด้วยขั้นตอนวิธี AES จะได้ข้อความ (TreeMsgNo) ซึ่งคั่นแต่ละค่าแอททริบิวต์ที่กำหนดหมายเลขแล้วด้วยเครื่องหมาย “|”

หลังจากถอดรหัสเอกสาร WSDL ส่วนที่ถูกเข้ารหัสแล้ว กลไก SWSDL ลบแท็ก <wsdl:cdescription> ออกจากเอกสาร WSDL

3.3.3 การเปรียบเทียบแอททริบิวต์ในเอกสาร WSDL (MatchDoc)

ขั้นตอนนี้เป็นการเปรียบเทียบค่าแอททริบิวต์ที่ได้จากการย่อยข้อความด้วยฟังก์ชันแฮชซึ่งเป็นข้อความย่อยที่แนบมากับเอกสาร WSDL ที่เข้ารหัสแล้ว (WSDLC) กับค่าแอททริบิวต์ที่สกัดได้จากข้อความ (TreeMsgNo) ในกระบวนการถอดรหัส (PDecrypt) โดยนำผลลัพธ์ที่ได้มาเปรียบเทียบกัน ถ้าได้ข้อความย่อยเดียวกัน แสดงว่าเอกสาร WSDL ที่ส่งมาจากฝั่งผู้ให้บริการไม่มีการปลอมแปลงเอกสาร ซึ่งขั้นตอนวิธีการเปรียบเทียบแอททริบิวต์แสดงดังภาพประกอบ 3.17

```
//Attr_num is attribute value that it is extracted from TreeMsgNo
// Attr_WSDL_H is attribute value that it is extracted from WSDLC
1.  Method MatchDoc(Attr_num,Attr_WSDL_H)
2.      Attr_H = H(Attr_num)
3.      If(Attr_H== Attr_WSDL_H)
4.          Remove number from Attr_num (Attr)
5.          Replace Attr_WSDL_H of WSDLC with Attr
6.      EndIf
7.  EndMethod
```

ภาพประกอบ 3.17 ขั้นตอนวิธีการเปรียบเทียบแอททริบิวต์ในเอกสาร WSDL

จากภาพประกอบ 3.17 การเปรียบเทียบค่าแอททริบิวต์ในเอกสาร WSDL มีขั้นตอนการทำงาน ดังนี้

1) ข้อความย่อย ($Attr_H$) ได้มาจากการสกัดค่าแอททริบิวต์จากข้อความที่ได้ในการถอดรหัส (TreeMsgNo) ซึ่งแต่ละค่าแอททริบิวต์มีหมายเลขกำหนด ($Attr_{num}$) มาผ่านกระบวนการแฮช

2) ข้อความย่อย ($Attr_{WSDL\_H}$) ได้จากการสกัดค่าแอททริบิวต์ในแท็ก <message> <portType> <service> แท็กย่อยภายในแท็กเหล่านี้ และแท็ก <operation> ภายใต้แท็ก <binding> จากเอกสาร WSDL ที่ถูกเข้ารหัสแล้ว (WSDLC)

3) กระบวนการเปรียบเทียบแอททริบิวต์คือการเปรียบเทียบข้อความย่อยที่สกัดได้ในขั้นตอนที่ 1 ($Attr_H$) และข้อความย่อยในขั้นตอนที่ 2 ($Attr_{WSDL\_H}$) ถ้าข้อความย่อยเป็นข้อความเดียวกันจะลบหมายเลขของค่าแอททริบิวต์ที่สกัดได้ในขั้นตอนที่ 1 ($Attr_{num}$) จะได้ค่า

แอททริบิวต์ต้นฉบับก่อนเข้ารหัส (Attr) แต่ถ้าข้อความย่อยไม่เหมือนกันแสดงว่าเอกสาร WSDL มีการปลอมแปลง

4) แทนที่ค่าแอททริบิวต์ที่ได้ในขั้นตอนที่ 3 (Attr) ในตำแหน่งค่าแอททริบิวต์เดิมที่สกัดได้ในขั้นตอนที่ 2 ($Attr_{WSDL\_H}$) ในเอกสาร WSDL ที่ถูกเข้ารหัส (WSDLC) จะได้เอกสาร WSDL ต้นฉบับ

ตัวอย่าง การเปรียบเทียบค่าแอททริบิวต์ในเอกสาร WSDL โดย “0 HelloWorldRequest|0.00 strName |0.01 xsd:string|0.10 strEmail|0.11 xsd:string” เป็นค่าแอททริบิวต์ในแท็ก <message> ซึ่งเป็นข้อความส่วนหนึ่งของ TreeMsgNo ที่ได้จากขั้นตอนการถอดรหัส (PDecrypt) และภาพประกอบ 3.18 เป็นตัวอย่างแท็ก <message> ซึ่งเป็นส่วนหนึ่งของ $WSDL_H$

```
<message name="10A488FFA001363427F0347AF9A69874386366BE">
        <part name="4763A4017C2938E5EB149536387DCF059A352F36"
        type="EE476447480D921A62A88DF4A6EA4A9B19674882"/>
        <part name="AFEF9BF391FBD1A8190057AA5550676F35EF3648"
        type="1035544FDAF3FE2DD348D65DEB23AB7DAAE122D3"/>
</message>
```

ภาพประกอบ 3.18 ตัวอย่างแท็ก <message> ของ $WSDL_H$

1. เมื่อสกัดแต่ละค่าแอททริบิวต์จาก TreeMsgNo จะได้

$Attr_{num}$ [0] = 0 HelloWorldRequest

$Attr_{num}$ [1] = 0.00 strName

$Attr_{num}$ [2] = 0.01 xsd:string

$Attr_{num}$ [3] = 0.10 strEmail

$Attr_{num}$ [4] = 0.11 xsd:string

2. ย่อยข้อความแต่ละค่าแอททริบิวต์ในขั้นตอนที่ 1 จะได้

$Attr_H$ [0] = 10A488FFA001363427F0347AF9A69874386366BE
$Attr_H$ [1] = 4763A4017C2938E5EB149536387DCF059A352F36
$Attr_H$ [2] = EE476447480D921A62A88DF4A6EA4A9B19674882
$Attr_H$ [3] = AFEF9BF391FBD1A8190057AA5550676F35EF3648
$Attr_H$ [4] = 1035544FDAF3FE2DD348D65DEB23AB7DAAE122D3

3. เปรียบเทียบข้อความย่อยในขั้นตอนที่ 2 กับค่าแอททริบิวต์ที่สกัดได้จากภาพประกอบ 3.18 ซึ่งข้อความย่อยที่ได้มีค่าแอททริบิวต์แต่ละค่าเหมือนกัน คือ

$Attr_H$ [0] = $Attr_{WSDL\_H}$ [0] = 10A488FFA001363427F0347AF9A69874386366BE
$Attr_H$ [1] = $Attr_{WSDL\_H}$ [1] = 4763A4017C2938E5EB149536387DCF059A352F36
$Attr_H$ [2] = $Attr_{WSDL\_H}$ [2] = EE476447480D921A62A88DF4A6EA4A9B19674882
$Attr_H$ [3] = $Attr_{WSDL\_H}$ [3] = AFEF9BF391FBD1A8190057AA5550676F35EF3648
$Attr_H$ [4] = $Attr_{WSDL\_H}$ [4] = 035544FDAF3FE2DD348D65DEB23AB7DAAE122D3

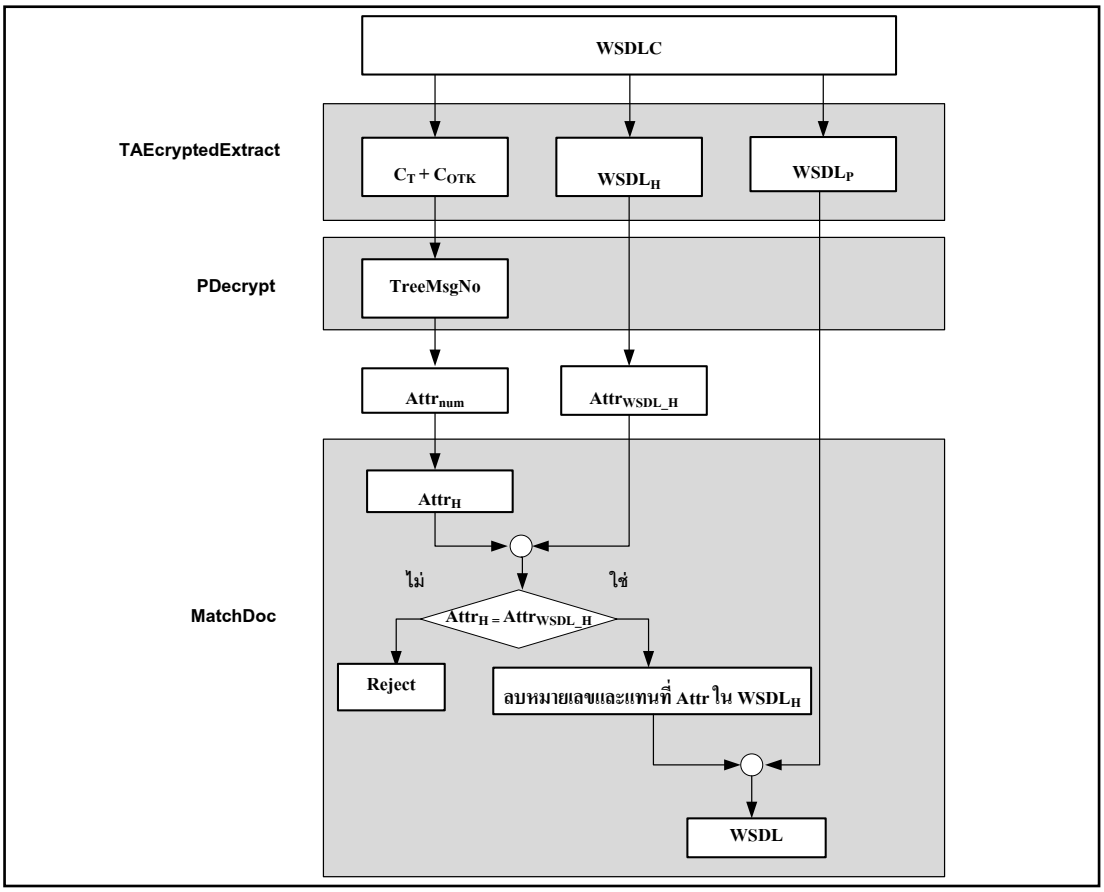
4. ลบหมายเลขของแต่ละค่าแอททริบิวต์ จะได้

Attr [0] = HelloWorldRequest
Attr [1] = strName
Attr [2] = xsd:string
Attr [3] = strEmail
Attr [4] = xsd:string

5. นำแต่ละค่าแอททริบิวต์ในขั้นตอนที่ 4 ไปแทนที่แต่ละตำแหน่งที่ตรงกันในแท็ก <message> ของเอกสาร $WSDL_H$ ดังภาพประกอบ 3.19

```
<message name="HelloWorldRequest">
        <part name="strName" type="xsd:string"/>
        <part name="strEmail" type="xsd:string"/>
</message>
```

ภาพประกอบ 3.19 ตัวอย่างการแทนที่แอททริบิวต์ในเอกสาร WSDL



ภาพประกอบ 3.20 ผลลัพธ์แต่ละขั้นตอนการถอดรหัสเอกสาร WSDL1.1

จากภาพประกอบ 3.20 ผลลัพธ์แต่ละขั้นตอนการถอดรหัสเอกสาร WSDL1.1 อธิบายดังนี้

ผลลัพธ์ของขั้นตอนการสกัดแท็กส่วนการเข้ารหัสเอกสาร WSDL (TAEcryptedExtract) จะประกอบด้วยข้อความไซเฟอร์ (C_T) ข้อความไซเฟอร์ของกุญแจ (C_{OTK}) เอกสาร WSDL ส่วนที่ไม่ได้เข้ารหัส ($WSDL_P$) และเอกสาร WSDL ส่วนที่ผ่านกระบวนการแฮช ($WSDL_H$)

ผลลัพธ์ของขั้นตอนการถอดรหัสเอกสาร (PDecrypt) คือข้อความไซเฟอร์ (C_T) และข้อความไซเฟอร์ของกุญแจ (C_{OTK}) โดยถอดรหัส C_T และ C_{OTK} ได้ข้อความ (TreeMsgNo)

ผลลัพธ์ของขั้นตอนการเปรียบเทียบแอททริบิวต์ในเอกสาร WSDL (MatchDoc) คือการนำค่าแอททริบิวต์ ($Attr_{num}$) ที่สกัดได้จาก TreeMsgNo มาผ่านกระบวนการแฮชได้ข้อความย่อย ($Attr_H$) เปรียบเทียบกับค่าแอททริบิวต์ ($Attr_{WSDL\_H}$) ซึ่งได้จากการสกัดข้อความย่อยในเอกสารส่วนที่ผ่านกระบวนการแฮช ($WSDL_H$) และถ้าข้อความย่อยเท่ากันลบหมายเลขจะได้ค่าแอททริบิวต์ (Attr) และนำไปแทนที่ข้อความย่อยในเอกสาร $WSDL_H$ และเมื่อรวมกับเอกสาร WSDL ส่วนที่ไม่ได้เข้ารหัส ($WSDL_P$) จะได้เอกสาร WSDL ต้นฉบับ

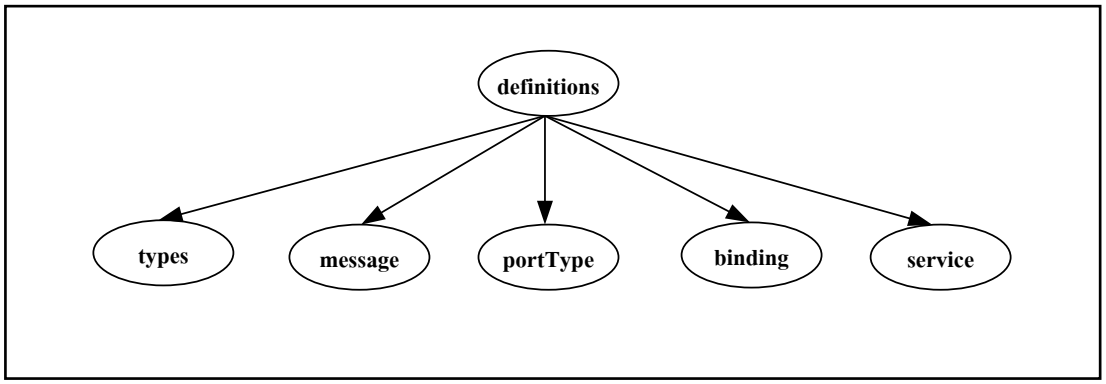
บทที่ 4

ประสิทธิภาพของกลไกและผลการทดลอง

วิทยานิพนธ์นี้ได้ออกแบบกลไกสำหรับความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส ดังรายละเอียดในบทที่ 3 ซึ่งในบทนี้จะนำเสนอชุดข้อมูลที่ใช้ในการทดลอง การออกแบบการทดลองที่ใช้ในการเปรียบเทียบกับระบบที่นำเสนอและผลลัพธ์ที่ได้จากการทดลอง โดยได้ทดลองกลไกที่นำเสนอเปรียบเทียบกับวิธีการปลอดภัยของ XML คือ XML Encryption และ XML Signature ซึ่งในการทดลองจะใช้ชุดข้อมูลของเอกสาร WSDL คือ Full_dataset (Hess, 2007 :Online)

4.1 ชุดข้อมูลที่ใช้ในการทดลอง

ชุดข้อมูลที่นำมาทดสอบต้องเป็นเอกสาร WSDL1.1 เท่านั้น โดยโครงสร้างของข้อมูลประกอบด้วยแท็กเหล่านี้ คือ แท็ก <definitions> แท็ก <types> แท็ก <message> แท็ก <portType> แท็ก <binding> และแท็ก <service> โดยภายในแท็กเหล่านี้จะประกอบด้วย “wsdl:” นำหน้าชื่อแท็กหรือไม่ก็ได้ ดังภาพประกอบที่ 4.1



ภาพประกอบ 4.1 โครงสร้างของเอกสาร WSDL1.1

4.2 การออกแบบการทดลอง

ในการทดลองเพื่อศึกษาประสิทธิภาพกลไกสำหรับความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส หรือเรียกว่า “SWSDL” ได้ทดลองโดยเปรียบเทียบประสิทธิภาพความปลอดภัยของเอกสาร WSDL กับวิธีการมาตรฐานความปลอดภัยของ XML คือ XML Signature และ XML Encryption หรือเรียกว่า “XMLSig_Enc” (Mirtalebi และ Khayyambashi, 2012) ดังอธิบายในงานวิจัยที่เกี่ยวข้องในบทที่ 2 ซึ่งแบ่งเป็น 2 ขั้นตอนการทำงาน คือ

1. ขั้นตอนการเข้ารหัสเอกสาร WSDL
2. ขั้นตอนการถอดรหัสเอกสาร WSDL

โดยในขั้นตอนการเข้ารหัสและถอดรหัสข้อมูลในเอกสาร WSDL วิธีการ XMLSig_Enc ใช้วิธีการเข้ารหัสข้อมูลแบบกุญแจสมมาตร AES เหมือนกันกับวิธีการ SWSDL ซึ่งการจับเวลาสำหรับการทดลองในขั้นตอนการเข้ารหัสและถอดรหัสเอกสาร WSDL จะเริ่มต้นตั้งแต่ขั้นตอนการอ่านเอกสาร WSDL จนถึงขั้นตอนเข้ารหัสเอกสาร WSDL เสร็จสิ้น สำหรับส่งเอกสาร WSDL ไปยังผู้ใช้บริการ หรือขั้นตอนการถอดรหัสเอกสาร WSDL เสร็จสิ้นสำหรับการนำเอกสาร WSDL ไปใช้งาน

4.3 ผลการทดลอง

ผลการทดลองในวิทยานิพนธ์นี้จะแบ่งเป็น 3 ส่วนได้แก่ การทดลองเปรียบเทียบประสิทธิภาพเวลาในการเข้ารหัสและถอดรหัสข้อมูลของเอกสาร WSDL โดยจำแนกแต่ละประเภทของข้อมูล การทดลองเปรียบเทียบประสิทธิภาพเวลาในการเข้ารหัสและถอดรหัสข้อมูลในเอกสาร WSDL โดยแบ่งตามขนาดเอกสาร การทดลองเปรียบเทียบความปลอดภัยของเอกสาร WSDL

4.3.1 ผลการทดลองเปรียบเทียบประสิทธิภาพเวลาในการเข้ารหัสและถอดรหัสข้อมูลของเอกสาร WSDL โดยจำแนกแต่ละประเภท

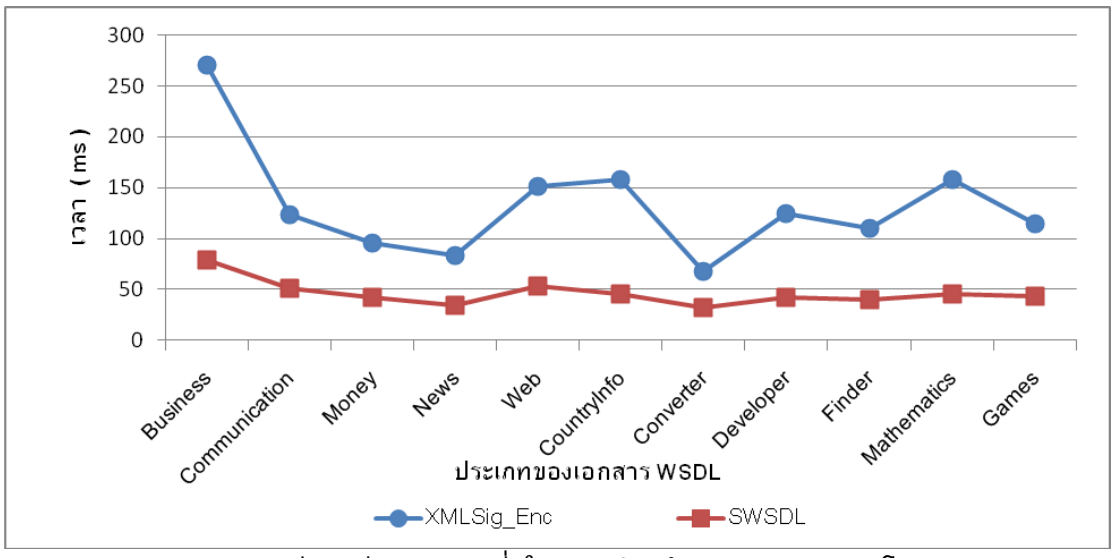
ในการทดลองเปรียบเทียบประสิทธิภาพเวลาในการเข้ารหัสและถอดรหัสเอกสาร WSDL ซึ่งเอกสาร WSDL มีขนาดเอกสารและข้อมูลแตกต่างกัน โดยทดลองกับเอกสาร WSDL จำนวน 396 เอกสาร ซึ่งจำแนกเอกสาร WSDL ได้เป็น 11 ประเภท แสดงดังตาราง 4.1 ดังนี้คือ

ตารางที่ 4.1 ประเภทข้อมูลของเอกสาร WSDL

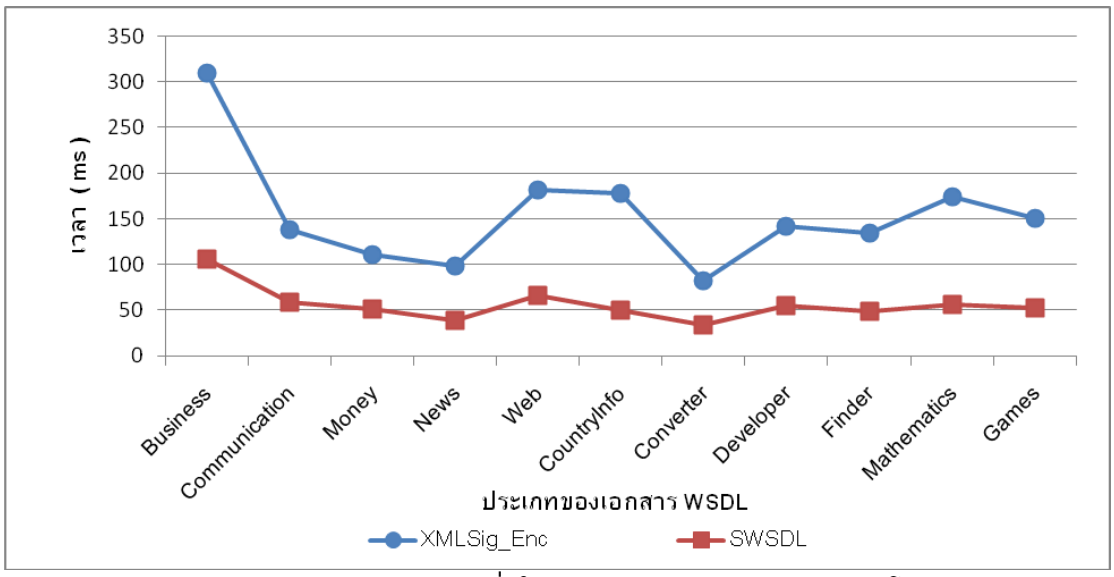
ประเภทเอกสาร	จำนวน (เอกสาร)	ขนาดเฉลี่ยของเอกสาร (KB)
Business	20	26.6
Communication	44	14.23
Money	56	13.1
News	28	6.37
Web	38	24.12
CountryInfo	63	12.95
Converter	49	6.53
Developer	36	15.24
Finder	43	12.24
Mathematic	10	8.57
Game	9	6.76

ตารางที่ 4.2 เวลาเฉลี่ยสำหรับการเข้ารหัสและถอดรหัสเอกสาร WSDL แต่ละประเภท

ประเภทของเอกสาร	เวลาเฉลี่ยสำหรับการเข้ารหัส (ms)		เวลาเฉลี่ยสำหรับการถอดรหัส (ms)	
	XMLSig_Enc	SWSDL	XMLSig_Enc	SWSDL
Business	270	78.75	310.2	105.4
Communication	123.39	51.34	138.57	59.11
Money	95.54	41.63	111.34	51.34
News	83.61	34.68	98.11	38.43
Web	151.4	53.34	182.21	66.18
CountryInfo	157.71	45.21	177.76	50.11
Converter	67.65	31.8	82.53	34.43
Developer	124.75	41.94	142.31	54.92
Finder	109.98	39.91	134.6	48.4
Mathematics	157.8	45.1	174.8	56.3
Games	114.56	46.67	150.89	51.89



ภาพประกอบ 4.2 เปรียบเทียบเวลาเฉลี่ยในการเข้ารหัสเอกสาร WSDL โดยจำแนกตามประเภทเอกสาร



ภาพประกอบ 4.3 เปรียบเทียบเวลาเฉลี่ยในการถอดรหัสเอกสาร WSDL โดยจำแนกตามประเภทเอกสาร

ผลการทดสอบแสดงให้เห็นว่าการเข้ารหัสและถอดรหัสเอกสาร WSDL ด้วยวิธีการ SWSDL ใช้ระยะเวลาในการเข้ารหัสและถอดรหัสเอกสาร WSDL เฉลี่ยน้อยกว่าวิธีการ XMLSig_Enc ในแต่ละประเภทเอกสาร WSDL คิดเป็น 63.10 % และ 63.13 % ตามลำดับ โดยทั่วไปการเข้ารหัสเอกสาร WSDL ใช้ระยะเวลาในการเข้ารหัสน้อยกว่าการถอดรหัส เนื่องจากเมื่อเข้ารหัสเอกสาร WSDL จะทำให้ขนาดของเอกสาร WSDL ใหญ่ขึ้นกว่าเดิม และ

เมื่อเอกสาร WSDL มีขนาดเพิ่มขึ้นจะทำให้ใช้เวลามากขึ้นด้วยในขั้นตอนการอ่านเอกสาร WSDL ซึ่งผลการทดสอบเวลาในการเข้ารหัสและถอดรหัสเอกสาร WSDL โดยจำแนกตามแต่ละประเภทแสดงดังตารางที่ 4.2 และผลการทดสอบเวลาในการเข้ารหัสและถอดรหัสเอกสาร WSDL แสดงดังภาพประกอบ 4.2 และ 4.3 ตามลำดับ

4.3.2 ผลการทดลองเปรียบเทียบประสิทธิภาพเวลาในการเข้ารหัสและถอดรหัสข้อมูลในเอกสาร WSDL โดยแบ่งตามขนาดเอกสาร

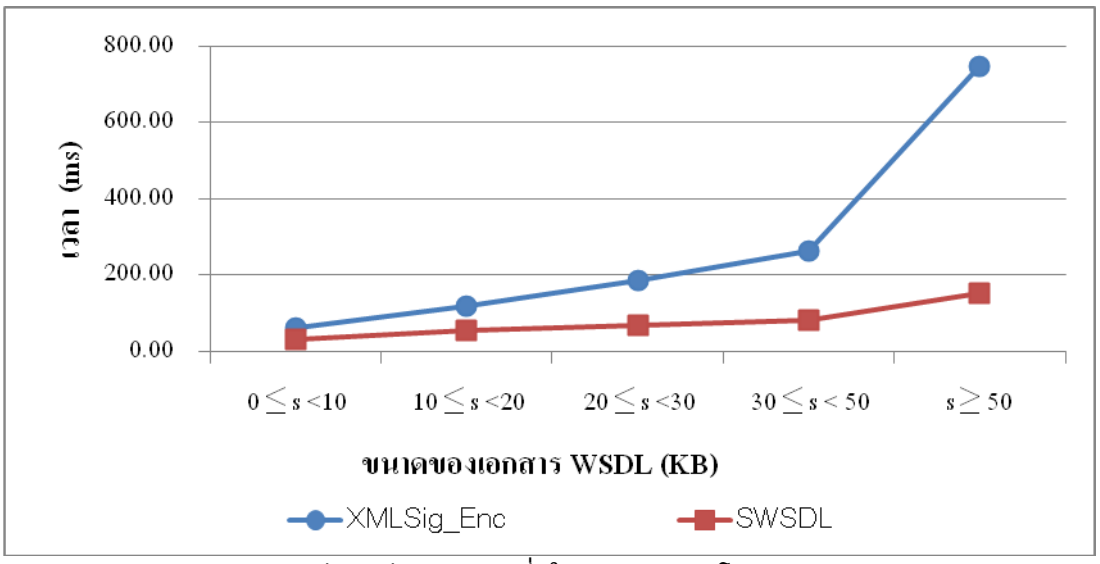
การทดลองเปรียบเทียบประสิทธิภาพเวลาในการเข้ารหัสและถอดรหัสข้อมูลในเอกสาร WSDL ซึ่งเอกสาร WSDL มีขนาดเอกสารแตกต่างกัน โดยขนาดเอกสาร WSDL ส่วนใหญ่เป็นขนาดเอกสาร WSDL ที่มีขนาดเล็ก มีขนาดไม่เกิน 10 KB ซึ่งขนาดเอกสาร WSDL ที่มีขนาดเล็กสุดและใหญ่สุดที่นำมาทดสอบ คือ 1.398 KB และ 215.621 KB ตามลำดับ ซึ่งการทดลองนี้ได้แบ่งขนาดเอกสาร WSDL ออกเป็น 5 ขนาด ดังตารางที่ 4.3

ตารางที่ 4.3 ขนาดข้อมูลเอกสาร WSDL

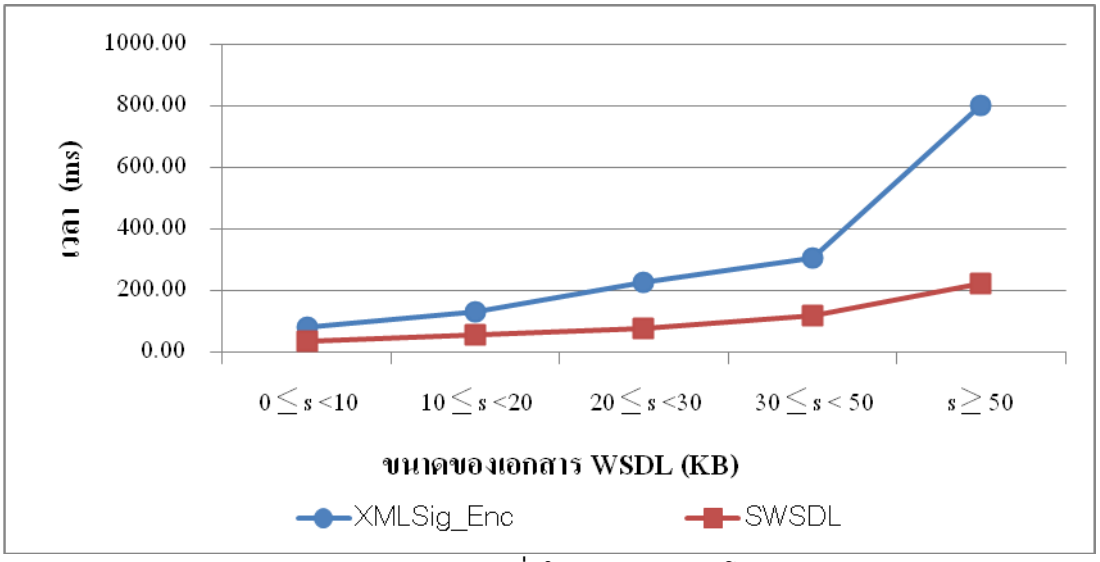
ขนาดเอกสาร WSDL (KB)	ขนาดเฉลี่ยของเอกสาร (KB)	จำนวน (เอกสาร)
$0 \leq s < 10$	4.63	267
$10 \leq s < 20$	14.72	66
$20 \leq s < 30$	24.82	24
$30 \leq s < 50$	37.78	18
$s \geq 50$	88.66	21

ตารางที่ 4.4 เวลาเฉลี่ยสำหรับการเข้ารหัสและถอดรหัสเอกสาร WSDL แต่ละขนาด

ขนาดเอกสาร WSDL (KB)	เวลาเฉลี่ยสำหรับการเข้ารหัส (ms)		เวลาเฉลี่ยสำหรับการถอดรหัส (ms)	
	XMLSig_Enc	SWSDL	XMLSig_Enc	SWSDL
$0 \leq s < 10$	62.64	29.49	78.53	32.97
$10 \leq s < 20$	117.55	53.11	132.05	56.33
$20 \leq s < 30$	185.21	67.5	227.54	75.46
$30 \leq s < 50$	262.28	79.89	306	119.56
$s \geq 50$	745.29	152.29	800.76	220.1



ภาพประกอบ 4.4 เปรียบเทียบเวลาเฉลี่ยในการเข้ารหัสโดยแบ่งตามขนาดเอกสาร



ภาพประกอบ 4.5 เปรียบเทียบเวลาเฉลี่ยในการถอดรหัสโดยแบ่งตามขนาดเอกสาร

ผลการทดสอบแสดงให้เห็นว่าการเข้ารหัสและถอดรหัสเอกสาร WSDL ด้วยวิธีการ SWSDL ใช้ระยะเวลาในการเข้ารหัสและถอดรหัสเอกสาร WSDL น้อยกว่าวิธีการ XMLSig_Enc โดยเมื่อขนาดเอกสาร WSDL มีขนาดใหญ่ขึ้นจะใช้เวลาในการเข้ารหัสและถอดรหัสเอกสาร WSDL เพิ่มขึ้นด้วย ผลการทดสอบเวลาในการเข้ารหัสและถอดรหัสเอกสาร WSDL แสดงดังตารางที่ 4.4 และจากภาพประกอบ 4.4 เปรียบเทียบเวลาเฉลี่ยในการเข้ารหัสโดยแบ่งตามขนาดเอกสาร และภาพประกอบ 4.5 เปรียบเทียบเวลาเฉลี่ยในการถอดรหัสโดยแบ่งตามขนาดเอกสาร ซึ่งขนาดเอกสาร WSDL มากกว่า 50 KB เป็นต้นไป ใช้เวลาในการเข้ารหัสและถอดรหัสเอกสาร WSDL มากเกินไป เมื่อเปรียบเทียบกับเวลาของขนาดเอกสาร

WSDL ขนาดอื่น เนื่องจากขนาดเอกสาร WSDL ในช่วงนี้ประกอบด้วยขนาดเอกสารค่อนข้างหลากหลาย คือ ขนาดเอกสาร ตั้งแต่ 50 KB จนถึงขนาด 215.621 KB ($50 \leq s \leq 215.621$ KB)

โดยสรุปแล้วเวลาในการเข้ารหัสเอกสาร WSDL ด้วยวิธีการ SWSDL ใช้ระยะเวลาในการเข้ารหัสและถอดรหัสเอกสาร WSDL น้อยกว่าวิธีการ XMLSig_Enc เนื่องจากวิธีการ SWSDL เข้ารหัสเอกสาร WSDL เพียงครั้งเดียว แต่วิธีการ XMLSig_Enc เข้ารหัสเอกสารหลายครั้งขึ้นอยู่กับจำนวนแท็ก <message> และ <portType> ที่ต้องการเข้ารหัสในเอกสาร WSDL

4.3.3 ผลการทดลองเปรียบเทียบความปลอดภัยของเอกสาร WSDL

จากการวิเคราะห์การโจมตีเอกสาร WSDL ดังอธิบายไว้ในบทที่ 2 การทดลองเข้ารหัสและถอดรหัสเอกสาร WSDL ด้วยวิธีการ SWSDL สามารถป้องกันการโจมตีเอกสาร WSDL ได้ดังต่อไปนี้

1) WSDL Scanning การโจมตีวิธีการนี้ส่วนใหญ่เกิดในส่วนของแท็ก <operation> ในแท็ก <portType> หรือแท็ก <binding> และแท็ก <message> เมื่อเข้ารหัสแท็กในส่วนนี้จะทำให้ผู้โจมตีไม่สามารถวิเคราะห์ได้ว่ามีเซอร์วิส หรือ Operation อะไรให้เรียกใช้งานบ้าง การเข้ารหัสด้วยวิธีการ SWSDL แสดงดังภาพประกอบ 4.6 (ก) และ (ข) ตามลำดับ

2) Parameter Tampering การโจมตีในส่วนนี้เกิดจากการเปลี่ยนแปลงพารามิเตอร์ในเอกสาร WSDL โดยการเปลี่ยนแปลงแท็ก <soap:address> ภายในแท็ก <service> เพื่อต้องการให้ผู้ใช้บริการเซอร์วิสนั้นเชื่อมต่อไปยังเซอร์วิสอื่นที่ผู้ไม่ประสงค์ดีได้จัดเตรียมไว้ ซึ่งเมื่อเข้ารหัสแท็กในส่วนนี้จะทำให้ผู้โจมตีไม่สามารถเปลี่ยนแปลงพารามิเตอร์นี้ได้ การเข้ารหัสด้วยวิธีการ SWSDL ในส่วนนี้แสดงดังภาพประกอบ 4.6 (ข)

```
1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns:tns="http://www.xmethods.net/sd/eBayWatcherService.wsdl"
    xmlns:xsd="http://www.w3.org/1999/XMLSchema" name="eBayWatcherService"
    targetNamespace="http://www.xmethods.net/sd/eBayWatcherService.wsdl">
3.  <message name="21D7D98219A3FDAA77B535E14F0229518C67CF0B">
4.          <part name="64769FF9D5404CA6A590002780CC4E71F353EB66"
                    type="EE476447480D921A62A88DF4A6EA4A9B19674882"/>
5.  </message>
6.  <message name="3B8E16AAB241E9FBC680C419B655E3F4152D0F62">
7.          <part name="1AEBAEEA1D0A07079091141F5FFF6D5FFD2E21A6"
    type="E678E3D44730AF6CBC274384E3994F744FCC6ECA"/>
8.  </message>
9.  <portType name="33B3186C50BDF3F60E5C055741BBBFADF8E6B85E">
10.         <operation name="30BBB01E28A9F4FAD3D02F2B869A8D103E4A765F">
11.         <input message="73793BF656675A078DE10CD2C3A918C28F255C10"
    name="AE2908D97637749ACD2541549A57090BFAEE33A1"/>
12.         <output message="D0A87848DA8E193A24BEA615539E985479F6866C"
    name="0880AE6FFAD2BDAF43045D424564EF8A1402A314"/>
13.        </operation>
14. </portType>
```

ภาพประกอบ 4.6 (ก) เอกสาร WSDL ที่เข้ารหัสด้วยวิธีการ SWSDL

```
15. <binding name="eBayWatcherBinding" type="tns:eBayWatcherPortType">
16.         <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
17.             <operation name="9BC4D439103DAFF4038E5F5D008A3EFD71A9D578">
18.                 <soap:operation soapAction=""/>
19.                  <input>
20.                       <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                         namespace="urn:xmethods-EbayWatcher" use="encoded"/>
21.                  </input>
22.                 <output>
23.                       <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                         namespace="urn:xmethods-EbayWatcher" use="encoded"/>
24.                 </output>
25.             </operation>
26.  </binding>
27.  <service name="BD58808682E5283DD0BF278AB8ACF08B9AB01FBA">
28.       <documentation>Checks current high bid for an eBay auction</documentation>
29.       <port binding="42ECA41227507D31AED6C5AD6564503171B468A2"
         name="9D76DB9B4BD30863C60A32F93D50422DFD0352E9">
30.        <soap:address location="592819413C57317304E7F163305F67FCA040279A"/>
31.       </port>
32.  </service>
33. <wsdl:cdescription>
34. <wsdl:algorithm>AES/128/PKCS5Padding</wsdl:algorithm>
35. <wsdl:key>19456E01C72046363B313FF683F814FE8351DB17B452F6F43C2AD0C0F6E4814C9146C
     4F1C994A515C1F518C2B355D1023F5E16195679A2764BE0407947D457A4</wsdl:key>
36. <wsdl:cipher>771E542828277E2512CDB7EDC85825EA0A0CE5EF736982E23FAA38CCF6BA1849B
     A8F4EEFB0FC28CB3ED5572F6C3F538615E24BC7AE835A2043EAA250CBC3638D3AED57C1AF0
     ABBC9FE3822D5D1130BE9C56998D3484710A0E08BE298898241F4924F862442A99F8268D9D10
     D365E665551AC72BC99554E842ACC3A29DFDBD9474A3A7C274B364CDF78710162DF4DE05AD
     8A61BDC05ABA7588E44449777E05828858DC08FF329A4197EA614A66962863ED5F2553F6511B
     EF77157D4209BA88A208E8E9C5129FE2193B7126AAFC7A5646D753C962612BF192016CB5B49A
     90B41BEFB3BC51B255ACE2C2EF0AAE6B7B933670E6FF909BACF5BCE51E7E9A4AEC71A1DC9
     53FE5FBC08ED4933324A3876CF9D7CCEF7AE6C8EB5AB1FF84DF745A37D80535B08AD2232D9
     6A97F1ADEAF982ED4A5B6B877621E6C9FA56DE09534D9622039FDD592CB1241577431CC18B9
     23E54EA3E1AD5BF6DECC8FAFA1E279E3683C1869ACCDC67EEC85CD0B6458B3992D7BEEC9
     A3ED37FFB53458FEF2DF6DA42267C6377587DECBDECD8862E455B439432D6FFDE14D4A37F3
     3EB5C8ADB57F2E96E22584AF02D95D354D283DE403D1CB651C3148B
37. </wsdl:cipher>
38. </wsdl:cdescription>
39. </definitions>
```

ภาพประกอบ 4.6 (ข) เอกสาร WSDL ที่เข้ารหัสด้วยวิธีการ SWSDL

```
1.  <?xml version="1.0" encoding="UTF-8"?><definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns:tns="http://www.xmethods.net/sd/eBayWatcherService.wsdl"
    xmlns:xsd="http://www.w3.org/1999/XMLSchema" name="eBayWatcherService"
    targetNamespace="http://www.xmethods.net/sd/eBayWatcherService.wsdl">
2.    <xenc:EncryptedData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
    Type="http://www.w3.org/2001/04/xmlenc#Element">
3.          <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
4.          <xenc:CipherData>
5.              <xenc:CipherValue>zvsjJm8dbgAeE47C8bB/ZSqdRF46AjuqTaFgicw4
                  GsvgwgOsicQ0HYUJTtf6/zXIWTnmvdbeWC8Ne0FUhOgzcyuDhsHbONaXeaiyhK/iq1LjtBb
    CPvDonitu6LKjFoL+THUf5nTqt3ef26hmdNOMObWuT1GlSgWb6mJco2RI9wQ=
6.              </xenc:CipherValue>
7.          </xenc:CipherData>
8.    </xenc:EncryptedData>
9.    <xenc:EncryptedData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
    Type="http://www.w3.org/2001/04/xmlenc#Element">
10.         <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
11.         <xenc:CipherData>
12.             <xenc:CipherValue>J4hHd0dlCK5dBUEUtkuk+ipyD04Z7R8J1zRt6SuSGYFYQRO4oC
                  f2tL4bVjwKufJtMoBwMV3CeKVaqtfYPm8FUoK6EeoBEnVls4fcGxw5DB35Ozax8nIJRd
                  HQsJtYO4x5+LPFBmg9Mq5EldNVRrWJcF/Otyy528i4Jo3HHDNJB4I=
13.             </xenc:CipherValue>
14.         </xenc:CipherData>
15.   </xenc:EncryptedData>
16.   <xenc:EncryptedData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
    Type="http://www.w3.org/2001/04/xmlenc#Element">
17.         <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
18.         <xenc:CipherData>
19.             <xenc:CipherValue>47hrg9lrjWLg8vMNDjfP2rAh0M37OaY9v45hliyB375nlOvebg8XeSfT
                  MlVjtZ0nKZJ/zjRtW9mZWuzQyPAlrDXpvLYWn7E3yK1fbxgM0dhrGZzkZ8lnpD/fW/edBVn
                  V/bzrFazUTSN4In4h1JocRndUjXC226pWkEEqxOJMkR/ykYSHO5fGo4p+C1wdYYm
                  Wo2Mh+U+iYLoslzjrURghiuXocAcJ4mYSF7yZS1dU56dMNdZVnBAlWVQw9s7joT
                  4J0bb3ly2f5ecoX7kCe49W4U895QUCDeVZnPR6L9dIdWNgqrDndhXWIZ/nhoRechxS
                 /MK8DMq7rz+Z6kge6toyY2 Hv9qLG3icYVAHxUGM73PCLG+VMwD0ElPA6gJz9K2hs
20.   </xenc:CipherValue>
21.       </xenc:CipherData>
22.   </xenc:EncryptedData>
```

ภาพประกอบ 4.7 (ก) ส่วนเข้ารหัส XML Encryption ด้วยวิธีการ XMLSig_Enc

```
23. <binding name="eBayWatcherBinding" type="tns:eBayWatcherPortType">
24.         <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
25.         <operation name="getCurrentPrice">
26.         <soap:operation soapAction=""/>
27.         <input>
28.         <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    namespace="urn:xmethods-EbayWatcher" use="encoded"/>
29.         </input>
30.         <output>
31.         <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    namespace="urn:xmethods-EbayWatcher" use="encoded"/>
32.         </output>
33.         </operation>
34. </binding>
35. <service name="eBayWatcherService">
36.         <documentation>Checks current high bid for an eBay auction</documentation>
37.         <port binding="tns:eBayWatcherBinding" name="eBayWatcherPort">
38.         <soap:address location="http://services.xmethods.net:80/soap/servlet/rpcrouter"/>
39.         </port>
40. </service>
41. <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
42. <SignedInfo>
43. <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
44. <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
45. <Reference URI=""><Transforms><Transform
    Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/></Transforms>
46. <DigestMethod
    Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/><DigestValue>P+YQJ3i6yQJeo9xBYNQAiv1H
    rEA=</DigestValue>
47. </Reference>
48. </SignedInfo>
49. <SignatureValue>jRBhZ2zHsoqIu9r04c05hEnEOSIoyLxQGiBc4l6HpH9UBH04PGmZyyl+BjwimhTGH
    DI/W7pQeQ8GnyBw1Ooccjwb74PiA37oExUcYh+K0CAXfY9SgiWNgg2uoPUzhjWLbOvAUQiRT7
    47BBEGTVfUfbpOZq9tLMwTujs/82uQ1zc=</SignatureValue><KeyInfo><KeyValue><RSAKeyValue>
    <Modulus>jfAd5uV38L36+lDZJrqfH9oLN86VJezXYfAeU+lrFoHlKAXVJLAi9hKvBHQRer4tPfdez6iSBK
    sl6IHkPnVRAKt0xU99uxi5QpymsWAX3qnBqHlw9Z70PwyZ+Xysfw4Q2tK2HtSgUOhMuaUcIf9sbHvf
    gbvcRPgxDZZqfIzDmDU=</Modulus><Exponent>AQAB</Exponent></RSAKeyValue></KeyValue>
    </KeyInfo>
50. </Signature>
51. </definitions>
```

ภาพประกอบ 4.7 (ข) ส่วนลงลายเซ็นดิจิทัล XML Signature ด้วยวิธีการ XMLSig_Enc

จากภาพประกอบ 4.7 (ก) และ (ข) แสดงส่วนการเข้ารหัส XML Encryption และ ส่วนลงลายเซ็นดิจิทัล XML Signature ด้วยวิธีการ XMLSig_Enc ตามลำดับ และการรักษาความปลอดภัยในการป้องกันการโจมตีเอกสาร WSDL มีดังนี้

1) WSDL Scannig วิธีการ XMLSig_Enc ไม่สามารถป้องกันการโจมตีนี้ได้เนื่องจากไม่ได้เข้ารหัสแท็ก <operation> ในแท็ก <binding> ในบรรทัดที่ 25 ดังรูปที่ 4.7 (ข) ซึ่งแท็กนี้เป็นแท็กที่ระบุชื่อเซอร์วิสที่เรียกใช้งานเช่นเดียวกับแท็ก <operation> ในแท็ก <portType> ซึ่งหากไม่ได้เข้ารหัสในแท็กนี้ด้วย จะทำให้ผู้ไม่ประสงค์ดีใช้ช่องโหว่นี้ในการโจมตีเอกสาร WSDL ได้ ซึ่งไม่ปลอดภัยเพียงพอเมื่อเทียบกับวิธีการ SWSDL

2) Parameter Tampering สามารถป้องกันการโจมตีวิธีการนี้ได้ โดยการลงลายเซ็นดิจิทัล XML Signature หากผู้ไม่ประสงค์ดีเปลี่ยนแปลงแท็ก <soap:address> ภายในแท็ก <service> เมื่อตรวจสอบความถูกต้องของเอกสาร WSDL ก่อนนำไปใช้งานจะสามารถตรวจสอบได้ว่าเอกสาร WSDL มีการปลอมแปลงหรือไม่

โดยสรุปแล้วการรักษาความปลอดภัยในการป้องกันการโจมตีเอกสาร WSDL วิธีการ SWSDL สามารถป้องกันการโจมตีแบบ WSDL Scanning และ Parameter Tampering ได้ส่วนวิธีการ XMLSig_Enc สามารถป้องกันการโจมตีแบบ Parameter Tampering ได้ในส่วนการโจมตีแบบ WSDL Scanning ยังมีช่องโหว่ให้ผู้ไม่ประสงค์ดีโจมตีเอกสาร WSDL ด้วยวิธีการนี้ได้ โดยแต่ละวิธีการใช้ขั้นตอนวิธีที่แตกต่างกัน สรุปได้ดังตารางที่ 4.5

ตารางที่ 4.5 เปรียบเทียบความปลอดภัยในการโจมตีเอกสาร WSDL

ประเภทการโจมตี	**วิธีการ**	
	SWSDL	**XMLSig_Enc**
WSDL Scanning	✓ (ใช้ขั้นตอนวิธีการเข้ารหัส)	✗ (XML Encryption แต่ไม่ได้เข้ารหัสแท็ก <operation> ในแท็ก <binding>)
Parameter Tampering	✓ (ใช้ขั้นตอนวิธีการย่อยข้อความ)	✓ (XML Signature)

บทที่ 5

บทสรุปและข้อเสนอแนะ

วิทยานิพนธ์นี้ได้ออกแบบกลไกสำหรับความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส (Mechanism for Securing WSDL Web Service) เพื่อทำให้เอกสาร WSDL มีความปลอดภัยในการนำไปใช้งานและช่วยลดระยะเวลาในการเข้ารหัสและถอดรหัสข้อมูล โดยการทำงานของกลไกสำหรับความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิสประกอบด้วย 2 ส่วนสำคัญคือ การเข้ารหัสเอกสาร WSDL สำหรับฝั่งผู้ให้บริการ และการถอดรหัสเอกสาร WSDL สำหรับฝั่งผู้ใช้บริการ นอกจากนี้ยังได้พัฒนาระบบเพื่อทดสอบกลไกการทำงานตามที่ได้ออกแบบไว้ ผลการศึกษาแสดงให้เห็นว่ากลไกดังกล่าวทำให้เอกสาร WSDL มีความปลอดภัยในการนำไปใช้งาน และใช้ระยะเวลาในการเข้ารหัสและถอดรหัสข้อมูลน้อยกว่าวิธีการที่นำมาเปรียบเทียบ

5.1 สรุปผลงานวิจัย

กลไกสำหรับความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส หรือเรียกว่า “SWSDL” ประยุกต์ใช้วิทยาการเข้ารหัสลับเพื่อให้ข้อมูลในเอกสาร WSDL มีความปลอดภัยในการนำไปใช้งาน และนำการย่อยข้อความซึ่งเป็นเทคนิคที่ใช้ตรวจสอบการมีบูรณภาพของข้อความ เพื่อตรวจสอบการปลอมแปลงเอกสาร WSDL มาประยุกต์ใช้ และการเข้ารหัสเอกสาร WSDL บางส่วนที่สำคัญเพื่อลดระยะเวลาในขั้นตอนการเข้ารหัสและถอดรหัสเอกสาร WSDL โดยเมื่อผู้ใช้บริการร้องขอเอกสาร WSDL และส่งกุญแจสาธารณะ ผู้ให้บริการจะเข้ารหัสเอกสาร WSDL ก่อนส่งไปให้ผู้ใช้บริการ และผู้ใช้บริการถอดรหัสเอกสาร WSDL ที่ได้รับด้วยกุญแจส่วนตัวของตนเองและนำเอกสาร WSDL นี้ไปเรียกใช้งานเว็บเซอร์วิส จากการทดลองที่ได้ออกแบบไว้และผลการทดลองในบทที่ 4 สามารถสรุปประเด็นได้ดังต่อไปนี้

5.1.1 ประเด็นเวลาการเข้ารหัสและถอดรหัส

วิธีการ SWSDL ใช้ระยะเวลาในการเข้ารหัสและถอดรหัสเอกสาร WSDL น้อยกว่าวิธีการ XMLSig_Enc เนื่องจาก SWSDL เข้ารหัสเอกสารเพียงครั้งเดียว แต่วิธีการ XMLSig_Enc เข้ารหัสเอกสารหลายครั้งขึ้นอยู่กับจำนวนแท็ก <message> และแท็ก <portType> ที่ต้องการเข้ารหัสในแต่ละเอกสาร WSDL

5.1.2 ประเด็นความปลอดภัย

1) วิธีการ XMLSig_Enc และวิธีการ SWSDL สามารถป้องกันการโจมตีแบบ Parameter Tampering ได้ โดยวิธีการ XMLSig_Enc ใช้วิธีการตรวจสอบความถูกต้องของเอกสารด้วยการลงลายเซ็นดิจิทัล XML Signature และวิธีการ SWSDL ใช้วิธีการตรวจสอบความถูกต้องของเอกสารโดยการย่อยข้อความหรือ ฟังก์ชันแฮช

2) วิธีการ SWSDL สามารถป้องกันการโจมตีแบบ WSDL Scanning ได้และมีประสิทธิภาพมากกว่าวิธีการ XMLSig_Enc เนื่องจากวิธีการ XMLSig_Enc ไม่ได้เข้ารหัสแท็ก <operation> ภายใต้แท็ก <binding> ทำให้ผู้ไม่ประสงค์ดีสามารถวิเคราะห์ได้ว่าเอกสาร WSDL นี้มีเซอร์วิสอะไรให้เรียกใช้งานบ้าง และสามารถที่จะใช้ช่องโหว่นี้ในการโจมตีเอกสาร WSDL นั้นได้

5.2 ปัญหาและอุปสรรค

1) เนื่องจากเว็บเซอร์วิสถูกพัฒนาด้วยภาษา XML ทำให้ต้องศึกษาภาษา XML ในการจัดการเอกสาร WSDL ได้แก่ XML Parser รวมถึงต้องศึกษาเกี่ยวกับโครงสร้างการทำงานของเว็บเซอร์วิส ความปลอดภัยของเว็บเซอร์วิสและมาตรฐานความปลอดภัยของ XML คือ XML Encryption และ XML Signature และการโจมตีของเว็บเซอร์วิสซึ่งมีหลายวิธีทำให้ใช้ระยะเวลาในการศึกษาค่อนข้างมาก

2) การป้องกันการโจมตีเว็บเซอร์วิสจำเป็นต้องใช้วิทยาการเข้ารหัสลับเพื่อเข้ารหัสและถอดรหัสข้อมูล ทำให้ต้องศึกษาขั้นตอนวิธีการต่างๆ เพื่อนำมาประยุกต์ใช้ในขั้นตอนการดำเนินการ ซึ่งวิธีการเหล่านี้มีความซับซ้อนจึงใช้เวลาค่อนข้างมาก

3) ข้อมูลที่นำมาใช้ในการทดสอบไม่มีข้อมูลที่กำหนดเป็นมาตรฐาน หรือ “Benchmark” ดังนั้นผู้วิจัยจึงใช้ชุดข้อมูลทดสอบนี้ (Hess, 2007: Online) ซึ่งเป็นชุดข้อมูลทดสอบที่ได้มีการรวบรวมเอกสาร WSDL แล้ว

5.3 ข้อเสนอแนะและงานวิจัยในอนาคต

กลไกสำหรับความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส เป็นกลไกที่ทำให้เอกสาร WSDL มีความปลอดภัยในการนำไปใช้งาน และป้องกันการโจมตีเอกสาร WSDL โดยการเข้ารหัสแท็กสำคัญเพียงบางส่วนในเอกสาร WSDL ทำให้เอกสาร WSDL เป็นความลับ เมื่อมีผู้ไม่ประสงค์ดีดักจับเอกสาร WSDL ก็จะไม่ทราบรายละเอียดข้อมูลสำคัญภายในเอกสาร WSDL เนื่องจากได้มีการเข้ารหัสไว้แล้ว ซึ่งยังมีประเด็นที่จะต้องพิจารณาเพื่อให้วิธีการที่นำเสนอมีประสิทธิภาพเพิ่มมากขึ้น ดังนี้

1) การทำให้ข้อมูลในเอกสารเป็นความลับโดยการเข้ารหัส ในขั้นตอนแรกผู้ใช้บริการต้องส่งกุญแจสาธารณะของตนเองมาด้วย ซึ่งงานวิจัยนี้ยังไม่ครอบคลุมในส่วนของการพิสูจน์ตัวตน คือ ไม่สามารถทราบได้ว่ากุญแจสาธารณะที่ส่งมานั้นเป็นกุญแจของผู้ขอใช้บริการจริงหรือไม่ อาจจะทำให้ผู้ไม่ประสงค์ดีใช้ช่องโหว่นี้ในการส่งกุญแจสาธารณะของตนเองเพื่อร้องขอเอกสาร WSDL

2) งานวิจัยนี้ได้ออกแบบและพัฒนาความปลอดภัยของเอกสาร WSDL1.1 ซึ่งปัจจุบันได้มีการกำหนดเอกสาร WSDL2.0 ขึ้น แต่ยังไม่มีการนำมาใช้งานอย่างแพร่หลาย หากมีการนำเอกสาร WSDL2.0 มาใช้งานมากขึ้น วิธีการที่นำเสนอควรจะครอบคลุมความปลอดภัยของเอกสาร WSDL2.0 ด้วย

3) สำหรับขั้นตอนการตรวจสอบความถูกต้องของเอกสาร WSDL วิธีการ SWSDL ประยุกต์ใช้การย่อยข้อความในแต่ละแอททริบิวต์ โดยจะย่อยข้อความหลายครั้งขึ้นอยู่กับจำนวนแอททริบิวต์ที่ต้องการเข้ารหัส ทำให้ใช้ระยะเวลานานในการเข้ารหัสและถอดรหัส

งานวิจัยในอนาคตสำหรับกลไกความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิสจึงมีแนวโน้มที่จะพัฒนาเพิ่มเติมตามประเด็นข้างต้นดังนี้ คือ

1) การพิสูจน์ตัวตนของผู้ขอใช้บริการเซอร์วิสก่อนที่จะส่งเอกสาร WSDL ที่เข้ารหัสแล้วกลับไป เพื่อเป็นการตรวจสอบตัวตนของบุคคลที่ร้องขอเอกสาร WSDL ทำให้ระบบกลไกสำหรับความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิสมีประสิทธิภาพและเอกสาร WSDL มีความปลอดภัยในการนำไปใช้งานมากขึ้น

2) การนำกลไกความปลอดภัยที่เสนอมาประยุกต์ใช้กับเอกสาร WSDL2.0 ซึ่งจำเป็นจะต้องวิเคราะห์ช่องโหว่ในการโจมตีเอกสาร WSDL2.0 เนื่องจากลักษณะโครงสร้างของเอกสาร WSDL1.1 และเอกสาร WSDL2.0 มีความแตกต่างกัน ซึ่งจะช่วยให้การใช้งานเว็บเซอร์วิสมีความปลอดภัยมากยิ่งขึ้น

3) ขั้นตอนการตรวจสอบความถูกต้องของเอกสาร WSDL ในการย่อยข้อความควรมีขั้นตอนวิธีการในการย่อยข้อความเพียงครั้งเดียว เพื่อลดระยะเวลาในการย่อยข้อความในขั้นตอนการเข้ารหัสและถอดรหัสเอกสาร WSDL ทำให้การเข้ารหัสและถอดรหัสเอกสาร WSDL ใช้ระยะเวลาน้อยลงจากวิธีการเดิม

บรรณานุกรม

กรมสรรพากร. 2012. กรมสรรพากรเปิดให้บริการด้าน Web Services ในระยะแรกให้บริการ 8 บริการ. [ออนไลน์] เข้าถึงได้จาก: http://www.rd.go.th/publish/42546.0.html. (วันที่สืบค้น 28 กุมภาพันธ์ 2557).

วิชุตา แก้วนพรัตน์. 2552. กลไกแจ้งสารสนเทศที่ปลอดภัยด้วยเทคโนโลยี RSS สำหรับอุปกรณ์สื่อสารเคลื่อนที่ที่รองรับโพรโทคอล TCP/IP, วิทยานิพนธ์วิทยาศาสตรมหาบัณฑิต คณะวิทยาศาสตร์ มหาวิทยาลัยสงขลานครินทร์ สงขลา.

Benz, B. and Durant J. R. 2003. XML Programming Bible. Wiley Publishing: NY.

CAPEC Common Attack Pattern Enumeration and Classification. 2013. CAPEC-95: WSDL Scanning (Version 2.3). https://capec.mitre.org/data/definitions/95.html. (accessed 10/7/2013).

Cerami, E. 2002. Web Service Essentials Distributed Applications with XML-RPC, SOAP, UDDI & WSDL. O'Reilly.

Coulouris, G., Dollimore, J., Kindberg, T. and Blair, G. 2012. DISTRIBUTED SYSTEMS Concepts and Design. Fifth Edition. Addison-Wesley: United Stated of America.

Dykes, L. and Tittel, E. 2005. XML for DUMMIES. 4. Wiley Publishing: NJ.

Fawcett, J. ,Quin, E. R. L. and Ayers,D. 2012. Beginning XML. 5. John Wiley & Sons: United States of America.

Hess, A. 2007. Collection of Categorized Web Services. http://www.andreas-hess.info/projects/annotator/ws2003.html. (accessed 20/11/2013).

Jensen M., Gruschka N. and Herkenhoner R. 2009. Jensen M., Gruschka N. and Herkenhoner R. 2009. A survey of attacks on web services Classification and countermeasures. Computer Science-Research and Development. 2009, 24(4): 185-197.

Mirtalebi, A. and Khayyambashi, M. 2011. Enhancing security of Web service against WSDL threats. Emergency Management and Management Sciences (ICEMMS), 2011 2nd IEEE International Conference. Beijing, August 8-10, 2011. pp. 920-923.

Mirtalebi, A. and Khayyambashi, M. R. 2012. A new Security Framework for Protecting WSDL File of Web Service. IJCSNS International Journal of Computer Science and Network Security. 2012, 12(9): pp.84-90.

Mogollon, M. 2007. Cryptography and Security Services: Mechanisms and Applications. Cybertech Publishing: United States of America.

Negm, W. 2004. Anatomy of a Web Services Attack a guide to Threats and Preventative Countermeasures. http://csis.org/files/media/csis/insights/anatomy_of_attack_wp.pdf. (accessed 7/7/2013).

Newcomer, E. 2002. Understanding Web Services XML, WSDL, SOAP, and UDDI. David Chappell.

Nordbotten N. A., 2009. XML and Web Services Security Standards. Communications Surveys & Tutorials, IEEE. 2009, 11(3): 4-21.

OWASP. 2013. Category:OWASP WebGoat Project. https://www.owasp.org/index.php/Category:OWASP_WebGoat_Project. (accessed 25/7/2013).

Rosenberg, J. and Remy, D. L. 2004. Securing Web Services with WS-Security. Sams Publishing: United States of America.

Shah, S. 2013. Web Services – Attacks and Defense. http://www.net-square.com/zip%20folders/WebServices_Info_Gathering.pdf. (accessed 11/7/2013).

Shahgholi, N., Mohsenzadeh, M., Seyyedi, M.A. and Qorani, S.H. 2011. A new security framework against Web services' XML attacks in SOA. Next Generation Web Services Practices (NWeSP), 2011 7th International Conference. Salamanca, October 19-21, 2011. pp. 314-319.

Sidharth, N. and Liu, J. 2007. IAPF: A Framework for Enhancing Web Services Security. Computer Software and Applications Conference, 2007. COMPSAC 2007. 31st Annual International. Beijing, July 24-27, 2007. pp.23-30.

Tran, K.T. 2013. Introduction to Web Services with Java. http://bookboon.com/en/introduction-to-web-services-with-java-ebook#download. (accessed 7/8/2013).

Wikipedia. 2014. Web Services Description Language. http://en.wikipedia.org/wiki/Web_Services_Description_Language. (accessed 23/7/2013).

WS-Attack.org. 2010. Metadata Spoofing. http://www.ws-attacks.org/index.php/WSDL_Parameter_Tampering. (accessed 15/7/2013).

XML Security. 2013. XML Digital Signature. http://www.xyzws.com/scdjws/SGS41/2. (accessed 3/10/2013).

ภาคผนวก

ภาคผนวก ก

ผลงานตีพิมพ์

เรื่อง	กลไกความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส
งานประชุมวิชาการ	การประชุมทางวิชาการระดับชาติ ด้านคอมพิวเตอร์และเทคโนโลยีสารสนเทศ ครั้งที่ 10
สถานที่	จังหวัดภูเก็ต ประเทศไทย
วันที่	8-9 พฤษภาคม 2557

กลไกความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส
A Mechanism for Securing WSDL Web Service

เจนจิรา หวังหลี (Janejira Wanglee)[1] และ ลัดดา ปรีชาวีรกุล (Ladda Preechaveerakul)[2]
ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยสงขลานครินทร์ วิทยาเขตหาดใหญ่
ตำบลคอหงส์ อำเภอหาดใหญ่ สงขลา 90110
[1]janeji.184@gmail.com, [2]ladda.p@psu.ac.th

บทคัดย่อ

ปัจจุบันมีการใช้งานเว็บเซอร์วิสเพิ่มมากขึ้นบนอินเทอร์เน็ต และการเรียกใช้งานเว็บเซอร์วิสนั้นจะมีเอกสาร WSDL (Web Service Description Language) ซึ่งเป็นเอกสารที่อธิบายวิธีการเรียกใช้งานเว็บเซอร์วิส ทำให้มีโอกาสเกิดการโจมตีเอกสาร WSDL ขึ้นเพื่อขโมยข้อมูล โดยทั่วไปเพื่อให้เอกสารมีความปลอดภัยจะเข้ารหัสเอกสาร WSDL ด้วยวิธีการมาตรฐานของ XML อย่างไรก็ตามการเข้ารหัสและถอดรหัสเอกสารด้วยวิธีนี้ใช้ระยะเวลาพอสมควร งานวิจัยนี้จึงนำเสนอกลไกความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส โดยการเข้ารหัสเอกสาร WSDL เพียงบางส่วน ซึ่งผลการทดลองแสดงให้เห็นว่ากลไกที่นำเสนอลดระยะเวลาในการเข้ารหัสได้ถึง 37.74% และลดระยะเวลาในการถอดรหัสได้ถึง 22.50% เมื่อเทียบกับงานวิจัยก่อนหน้านี้

คำสำคัญ: XML WSDL ความปลอดภัย เว็บเซอร์วิส

Abstract

At present, web services are widely used over the Internet. Since Web Service Description Language (WSDL) is used to describe the public interface to a specific web service, it is possible to be attacked from an unauthorized user. Generally, a WSDL document is encrypted using XML security standards. However, it spends times to encrypt and decrypt the WSDL document. This paper then, proposes a mechanism for securing WSDL documents in web service (SWSDL) using partial encryption in the WSDL file. The experimental result shows that SWSDL can reduce the encryption time to 37.74% and the decryption time to 22.50% in comparison with a previous research.

Keywords: XML, WSDL, Security, Web service

1.บทนำ

ปัจจุบันนี้มีการนำเว็บเซอร์วิสมาประยุกต์ใช้งานเพื่อตอบสนองความต้องการทางด้านธุรกิจ ซึ่งเว็บเซอร์วิสสนับสนุนการทำงานระหว่างกัน (Interoperation) โดยใช้ภาษา XML [1] ในการรับส่งข้อมูลระหว่างผู้ให้บริการและผู้ใช้บริการ ดังนั้นหลายองค์กรจึงหันมาให้ความสำคัญกับการใช้งานเว็บเซอร์วิสมากขึ้น อย่างไรก็ตามการใช้งานเว็บเซอร์วิส ผู้ใช้บริการเซอร์วิสนั้นๆจำเป็นจะต้องรู้วิธีการติดต่อกับผู้ให้บริการ ซึ่งวิธีการติดต่อเรียกใช้งานเซอร์วิสอธิบายในเอกสาร WSDL (Web Service Description Language) [2] การใช้งานเว็บเซอร์วิสโดยทั่วไปไม่มีกลไกสำหรับความปลอดภัยเพื่อเผยแพร่ข้อมูลเอกสาร WSDL ผ่านเครือข่ายอินเทอร์เน็ตจากผู้ให้บริการไปยังผู้ใช้บริการ ซึ่งเอกสาร WSDL อาจถูกโจมตีได้ แม้ว่า XML จะมี XML Encryption [3] และ XML Signature [3] เป็นมาตรฐานรองรับความปลอดภัยในส่วนนี้แล้วก็ตาม แต่ด้วยขั้นตอนวิธีที่ยุ่งยากซับซ้อนและใช้เวลาพอสมควรในการเข้ารหัสและถอดรหัสข้อมูลเพียงบางส่วนเพื่อต้องการปกปิดข้อมูลในเอกสาร WSDL ให้เป็นความลับ บทความนี้จึงเสนอกลไกสำหรับความปลอดภัยของเอกสาร WSDLโดยการเข้ารหัสเอกสาร WSDL เพียงบางส่วน เพื่อปกปิดข้อมูล และป้องกันการปลอมแปลงเอกสาร WSDL โดยผู้ใช้บริการถอดรหัสเอกสาร WSDL ก่อนเรียกใช้เว็บเซอร์วิสนั้น กลไกดังกล่าวทำให้ข้อมูลเอกสาร WSDL เป็นความลับและมีความปลอดภัยเมื่อส่งผ่านระบบเครือข่ายอินเทอร์เน็ตจากผู้ให้บริการไปยังผู้ใช้บริการ ป้องกันการโจมตีเอกสาร WSDL และลดระยะเวลาในการเข้าและถอดรหัสข้อมูล ทำให้การใช้งานเว็บเซอร์วิสมีความปลอดภัยและน่าเชื่อถือ

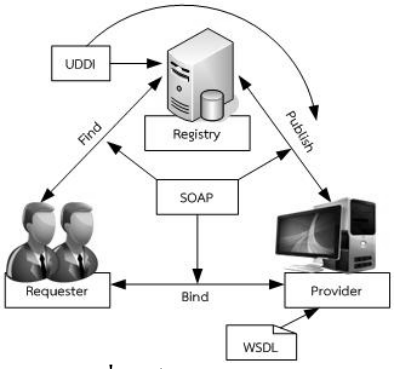
เนื้อหาของบทความในส่วนที่ 2 กล่าวถึงทฤษฎีและงานวิจัยที่เกี่ยวข้อง ส่วนที่ 3 อธิบายการทำงานของกลไกความปลอดภัย

ของเอกสาร WSDL ในส่วนที่ 4 แสดงการพัฒนากลไกความปลอดภัยและผลลัพธ์ และส่วนที่ 5 เป็นบทสรุป

2. ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 เว็บเซอร์วิส

เว็บเซอร์วิส [4] ประกอบ ด้วย WSDL SOAP (Simple Object Access Protocol) และ UDDI (Universal Description Discovery and Integration) แสดงดังภาพที่ 1 โดยมีการทำงานร่วมกันดังนี้



ภาพที่ 1: พื้นฐานเว็บเซอร์วิส [2]

ผู้ให้บริการ .1(Provider) เตรียมบริการเซอร์วิสให้เรียกใช้และเผยแพร่เซอร์วิสที่จัดทำขึ้นซึ่งก็คือ เอกสาร WSDL ไปยัง Registry หรือ UDDI

2. UDDI ให้บริการลงทะเบียน (Registry) เซอร์วิสและค้นหาเซอร์วิส UDDI เป็นบริการเก็บรวบรวมเซอร์วิสที่ผู้ให้บริการประกาศไว้

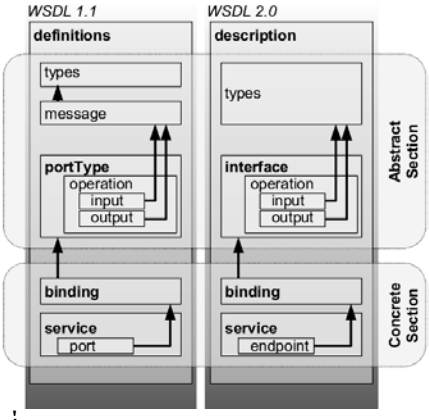
3. ผู้ขอใช้บริการ (Requester) เป็นผู้เรียกใช้เซอร์วิส ค้นหาเซอร์วิสจาก Registry แล้วเรียกใช้เซอร์วิสจากผู้ให้บริการ ซึ่งใช้ SOAP เป็นโพรโทคอลในการแลกเปลี่ยนข้อมูลระหว่างกันบนเว็บเซอร์วิส

องค์กร W3C [5] เป็นผู้กำหนดมาตรฐานเอกสาร WSDL ซึ่งเป็นเอกสารที่อยู่ในรูปแบบของภาษา XML เพื่ออธิบายการเรียกใช้งานเว็บเซอร์วิส ตัวอย่างและคำอธิบายแต่ละแท็กของเอกสาร WSDL 1.1 แสดงดังตารางที่ 1

ตารางที่ 1: ตัวอย่างแท็กของเอกสาร WSDL1.1

WSDL tag	คำอธิบาย
<definitions>	กำหนดชื่อของเซอร์วิส
<types>	อธิบายชนิดข้อมูล
<message>	อธิบายข้อมูลที่ใช้ในการแลกเปลี่ยน
<portType>	กำหนดฟังก์ชันเรียกใช้งาน
<binding>	อธิบายโพรโทคอล
<service>	กำหนดที่อยู่ของเซอร์วิส

ปัจจุบัน W3C ได้ระบุข้อกำหนดเกี่ยวกับ WSDL2.0 ซึ่งมีข้อแตกต่างจาก WSDL1.1 ดังภาพที่ 2



ภาพที่ 2: เปรียบเทียบเอกสาร WSDL 1.1 กับ WSDL 2.0 [6]

2.2 การโจมตีเอกสาร WSDL

ปัจจุบันนี้มีการโจมตีเอกสาร WSDL [7, 8] แบ่งเป็น 2 ประเภทหลักๆคือ 1) *WSDL Scanning* เป็นการโจมตีโดยผู้โจมตี (Hacker) อ่านแล้ววิเคราะห์เอกสาร WSDL และหาช่องโหว่ในเอกสาร WSDL เพื่อดึงข้อมูลจากเว็บเซอร์วิส *2) Parameter Tampering* เป็นการใช้งานเอกสาร WSDL ผิดรูปแบบ โดยผู้โจมตีส่งพารามิเตอร์ซ้ำๆกันหลายครั้ง แล้วหารูปแบบพารามิเตอร์ที่แตกต่างกันในการเข้าถึงข้อมูลที่ไม่ได้รับอนุญาต เช่น การใส่อักขระพิเศษเครื่องหมาย ‘ หรือ % เป็นต้น

ซึ่งการโจมตีทั้ง 2 ประเภทนี้โดยส่วนใหญ่จะโจมตีในแท็ก <message> แท็ก <portType> และแท็ก <service> ในเอกสาร WSDL

2.3 งานวิจัยที่เกี่ยวข้อง

เว็บเซอร์วิสถูกนำมาใช้งานอย่างแพร่หลายในด้านธุรกิจ การทำธุรกรรมทางด้านการเงิน เมื่อมีผู้ประสงค์ร้ายต้องการโจรกรรมข้อมูล จึงเกิดการโจมตีเว็บเซอร์วิสในส่วนของเอกสาร WSDL และได้มีงานวิจัยที่พัฒนาวิธีการป้องกันการโจมตีเอกสาร WSDL ในเว็บเซอร์วิส

ปี ค.ศ .2007 Sidharth และ Liu [9] ได้เสนอวิธีการจำกัดผู้ใช้ในการเข้าถึงเอกสาร WSDL และเอกสาร WSDL ไม่มีข้อมูลที่เป็นประโยชน์ต่อผู้โจมตี

ปี ค.ศ.2011 Mirtalebi และ Khayyambashi [10] ได้เสนอวิธีการเข้ารหัสแท็ก <message> และแท็ก <portType> ด้วย XML Encryption โดยใช้กุญแจสมมาตรหรือกุญแจอสมมาตรขึ้นอยู่กับความเหมาะสมในการนำเว็บเซอร์วิสไปใช้งานและปี ค.ศ 2012 [11] ได้พัฒนาโดยนำ XML Signature มาประยุกต์เพื่อให้ WSDL ปลอดภัยมากขึ้น

Shahgholi และคณะ [12] ได้เสนอการเข้ารหัสเอกสาร WSDL ด้วย XML Signature และ XML Encryption และ PKI (Public Key Infrastructure) [3] ในปี ค.ศ. 2011

อย่างไรก็ตามงานวิจัยดังกล่าวข้างต้น มีวิธีการเข้ารหัสที่ยุ่งยากซับซ้อนและใช้ระยะเวลาในการเข้ารหัสพอสมควร ดังนั้นงานวิจัยนี้จึงเสนอกลไกความปลอดภัยของเอกสาร WSDL โดยการเข้ารหัสเอกสาร WSDL เพียงบางส่วนเพื่อลดระยะเวลาการทำงาน และป้องกันการโจมตีเอกสาร WSDL

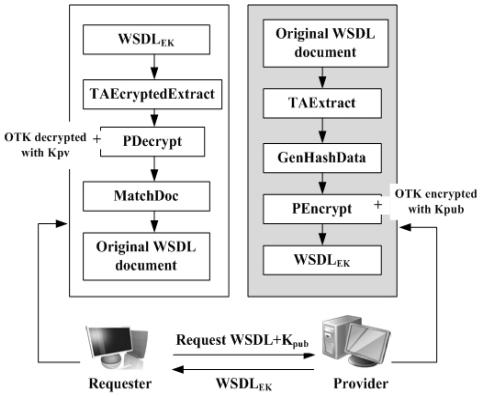
3. กลไกความปลอดภัยของเอกสาร WSDL

3.1 แนวคิดสำหรับการออกแบบ

เพื่อให้เอกสาร WSDL มีความปลอดภัยสำหรับการนำไปใช้งาน งานวิจัยนี้จึงเสนอกลไกการเข้ารหัสและถอดรหัสเอกสารเพียงบางส่วนเพื่อลดระยะเวลา นอกจากนี้ผู้ใช้บริการยังได้รับเอกสาร WSDL ที่เป็นความลับและถูกต้องสมบูรณ์จากผู้ให้บริการ

3.2 กลไกความปลอดภัยของเอกสาร WSDL (SWSDL)

การทำงานของกลไกความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส ประกอบด้วยส่วนสำคัญ 2 ส่วน คือ ส่วนเข้ารหัสเอกสาร WSDL สำหรับผู้ให้บริการ และส่วนถอดรหัสเอกสาร WSDL สำหรับผู้ใช้บริการ ซึ่งกลไกความปลอดภัยของเอกสาร WSDL อธิบายได้ดังภาพที่ 3



ภาพที่ 3: ขั้นตอนของกลไกความปลอดภัยของเอกสาร WSDL(SWSDL)

จากภาพที่ สามารถอธิบายขั้นตอนการทำงานของแต่ละ 3 ส่วนได้ดังนี้

3.2.1. ส่วนเข้ารหัสเอกสาร สำหรับผู้ให้บริการ

เมื่อผู้ใช้บริการ (Requester) ร้องขอเอกสาร WSDL และส่งกุญแจสาธารณะของตนเองกับผู้ให้บริการ (Provider) ดังนั้นจะเข้าสู่ส่วนเข้ารหัสเอกสาร WSDL โดยมีขั้นตอนคือ 1) สกัดค่าแอททริบิวต์ในเอกสาร WSDL (TAExtract) โดยจะสกัดเพียงบางแท็ก คือ แท็ก <message> แท็ก <portType> และ แท็ก <service> 2) กำหนดหมายเลขแต่ละค่าแอททริบิวต์ที่สกัดได้และนำมาผ่านกระบวนการแฮช (GenHashData) เพื่อย่อยข้อความ 3) เข้ารหัสค่าแอททริบิวต์ที่ผ่านกระบวนการกำหนดหมายเลขแล้วโดยใช้กุญแจสมมาตรที่ใช้เพียงครั้งเดียว (One-Time Key: OTK) และเข้ารหัสกุญแจสมมาตรด้วยกุญแจสาธารณะ) K_{pub} (ของผู้ใช้บริการ (PEncrypt)

จากนั้นผู้ให้บริการส่งเอกสาร WSDL ที่เข้ารหัสแล้ว ($WSDL_{EK}$) ซึ่งแนบกุญแจที่ใช้ในการเข้ารหัสไว้ภายในเอกสาร WSDL ไปยังผู้ใช้บริการ

3.2.2. ส่วนถอดรหัสเอกสาร สำหรับผู้ใช้บริการ

ผู้ใช้บริการถอดรหัสเอกสาร $WSDL_{EK}$ โดยมีขั้นตอนการทำงานคือ 1) สกัดแท็กในส่วนของการเข้ารหัสเอกสาร $WSDL_{EK}$ (TAEcryptedExtract) 2) ถอดรหัส OTK ด้วยกุญแจ

ส่วนตัว (K_{pv}) ของผู้ใช้บริการ และใช้กุญแจนี้ในการถอดรหัสเอกสาร $WSDL_{EK}$ (PDecrypt) 3) นำค่าแอททริบิวต์ที่ได้จากการถอดรหัสผ่านกระบวนการแฮชมาเปรียบเทียบกับค่าแอททริบิวต์ที่สกัดได้จากเอกสาร $WSDL_{EK}$ (MatchDoc) ถ้าผลลัพธ์ที่ได้เท่ากัน ผู้ใช้บริการจะนำเอกสาร WSDL ต้นฉบับไปเรียกใช้งานเว็บเซอร์วิสต่อไป

4. การพัฒนากลไกความปลอดภัยและผลลัพธ์

สำหรับกลไกความปลอดภัยของเอกสาร WSDL พัฒนาด้วยภาษาจาวา และประกอบด้วยการทำงานที่สำคัญ ส่วน คือ ส่วน 2 การเข้ารหัสและส่วนการถอดรหัส ภาพที่ 4แสดงขั้นตอนวิธีการเข้ารหัส

Algorithm1: Encryption data (WSDL)

1. Extract each attribute value (Attr) in <message>, <portType> and <service> tag
2. **For** each Attr
3. Assign number in each Attr ($Attr_{num}$)
4. Hash_Value = H($Attr_{num}$)
5. Replace Attr with Hash_Value
6. Concatenate $Attr_{num}$ with "|" symbol (Msg)
7. **Endfor**
8. Encrypt Msg with OTK
9. Encrypt OTK with K_{pub}
10. Create <wsdl:cdescription> tag
11. Insert cipher msg in <wsdl:cipher> tag
12. Insert cipher key in <wsdl:key> tag

ภาพที่ 4: ขั้นตอนวิธีการเข้ารหัส

จากขั้นตอนวิธีการเข้ารหัสดังภาพที่ 4 ประกอบด้วยขั้นตอนวิธีย่อย 2 วิธีคือ

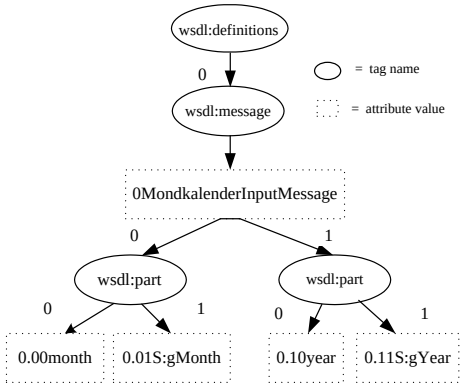
1. ขั้นตอนการกำหนดหมายเลขและแฮช (GenHashDoc)

เป็นการกำหนดหมายเลขให้แต่ละค่าแอททริบิวต์ โดยเอกสาร WSDL จะถูกแปลงให้อยู่ในรูปของโครงสร้างต้นไม้แล้วจึงกำหนดหมายเลขค่าแอททริบิวต์จากบนลงล่างและจากซ้ายไปขวาตามลำดับ ทำให้แต่ละค่าแอททริบิวต์มีหมายเลขไม่ซ้ำกัน เพื่อป้องกันค่าแอททริบิวต์ที่เหมือนกันเมื่อผ่านกระบวนการแฮช

```
<wsdl:definitions>
<wsdl:message name="MondkalenderInputMessage">
  <wsdl:part name="month" type="s:gMonth"/>
  <wsdl:part name="year" type="s:gYear"/>
</wsdl:message>
</wsdl:definitions>
```

ภาพที่ 5 : ตัวอย่างแท็ก <wsdl:message> ของเอกสาร WSDL1.1

จากตัวอย่างแท็กในภาพที่ 5 เอกสาร WSDL จะนำมาแปลงเป็นรูปแบบโครงสร้างต้นไม้และกำหนดหมายเลขค่าแอททริบิวต์ (*Attr_Id และ Attr_Value*) ดังภาพที่ 6 นำแต่ละแอททริบิวต์ที่สร้างขึ้นมาใหม่มาผ่านกระบวนการแฮช ได้ *Hash_Value* และนำไปแทนที่ค่าแอททริบิวต์เดิมในเอกสารWSDL ดังภาพ 7



ภาพที่ 6 : รูปแบบโครงสร้างต้นไม้ของแท็ก <wsdl:message>

```
<wsdl:message name="CBAE4DA3FF4C70B3032A6B6
8879F00661CECE4DD">
 <wsdl:part name="F74F4A7582C301F4B2394398DBF2
1E17B4DC1525" type="8A31279A6C28921CB9678487
3FDF2BA3C28C242E"/>
  <wsdl:part name="DA92C4644F8E821948A290D7E7
FD1942B523EC7D" type="119706A849E07067AC74E2
E7FDE0CB73DF05EC3A"/> </wsdl:message>
```

ภาพที่ 7: เอกสาร WSDL ที่ผ่านกระบวนการแฮช

2. ขั้นตอนการเข้ารหัสเอกสาร WSDL (PEncrypt)

เป็นการนำข้อความที่ได้จากการรวมแต่ละค่าแอททริบิวต์ที่กำหนดหมายเลขแล้วมาเข้ารหัสโดยใช้ OTK ด้วยขั้นตอนวิธี AES และเข้ารหัส OTK โดยใช้กุญแจสาธารณะของผู้ใช้บริการด้วยขั้นตอนวิธี RSA โดยกำหนดให้เอกสาร $WSDL_{EK}$ ระบุส่วน

ที่เข้ารหัสในเอกสารWSDLในแท็ก <wsdl:cdescription> โดยมีรูปแบบดังแสดงในภาพที่ 8 และตัวอย่างแสดงดังภาพที่ 9

```
<wsdl:cdescription>
<wsdl:algorithm> Encryption Algorithm</wsdl:algorithm>
<wsdl:key>Cipher Key</wsdl:key>
<wsdl:cipher>Cipher Message </wsdl:cipher>
</wsdl:cdescription>
```

ภาพที่ 8: รูปแบบแท็ก <wsdl:cdescription> ของเอกสาร WSDL

จากภาพที่ 8 กำหนดให้

<wsdl:cdescription> คือ ข้อมูลที่จำเป็นเกี่ยวกับการเข้ารหัส

<wsdl:algorithm> คือ ขั้นตอนวิธีการเข้ารหัส

<wsdl:key> คือ ข้อความไซเฟอร์ของกุญแจ

<wsdl:cipher> คือ ข้อความไซเฟอร์ของเอกสาร WSDL

```
<wsdl:cdescription>
<wsdl:algorithm>AES/128/PKCS5Padding</wsdl:algorith
m><wsdl:key>43d5537e82eca141f2a1b8088298c6a9ef
</wsdl:key><wsdl:cipher>ba54740f5b7813ee51a3e229c8c
23d1cabe24af906e132a442f4ffbf4a12f90872225aefc45ce4
c8e47cd6faa079fce40508b59ed9efc295a4c307af65ffc6ca3
0b4f80ba996833f39b25a92122ae22172e47bfc0e99cc04c4
bf8869d95b0074d6176ea9acb840aff1dae2a8fc802f8d0b0e
6</wsdl:cipher></wsdl:cdescription>
```

ภาพที่ 9: ตัวอย่างแท็ก <wsdl:cdescription> ในการเข้ารหัส WSDL

Algorithm2: Decryption data ($WSDL_{EK}$)

1. Extract <wsdl:cdescription> from $WSDL_{EK}$
2. Extract <wsdl:key> in <wsdl:cdescription> tag
3. Decrypt cipher key with K_{pv} (OTK)
4. Extract <wsdl:cipher> in <wsdl:cdescription> tag
5. Decrypt cipher msg with OTK (Msg_{DK})
6. Delete <wsdl:cdescription> tag from $WSDL_{EK}$
7. Extract each Attr in <message>, <portType> and <service> of $WSDL_{EK}$ ($Attr_{EK}$)
8. **For** each $Attr_{EK}$
9. Extract each Attr of Msg_{DK} ($Attr_{DK}$)
10. MatchDoc ($Attr_{DK}$, $Attr_{EK}$)
11. **Endfor**

ภาพที่ 10 : ขั้นตอนวิธีการถอดรหัส

สำหรับการถอดรหัสเอกสาร เมื่อผู้ใช้บริการได้รับเอกสาร $WSDL_{EK}$ จะถอดรหัสเอกสารด้วยอัลกอริทึมดังภาพที่ 10 โดยมีขั้นตอนย่อยที่สำคัญภายใน ส่วน 2คือ

1. ขั้นตอนการถอดรหัสเอกสาร WSDL (PDecrypt)

สกัดแท็กที่ระบุส่วนเข้ารหัส แล้วจึงถอดรหัสเอกสาร $WSDL_{EK}$ โดยถอดรหัสกุญแจสมมาตร (OTK) ที่ถูกเข้ารหัสด้วยกุญแจส่วนตัวของผู้ใช้บริการ และใช้กุญแจ OTK นี้ในการถอดรหัสเอกสาร $WSDL_{EK}$ จะได้ค่าแอททริบิวต์ที่คั่นด้วยสัญลักษณ์ "|"

2. ขั้นตอนการจับคู่ค่าแอททริบิวต์ (MatchDoc)

เมื่อสกัดแต่ละค่าแอททริบิวต์ที่คั่นด้วยสัญลักษณ์ "|" แล้วนำมาเปรียบเทียบกับค่าแอททริบิวต์ใน $WSDL_{EK}$ ดังขั้นตอนวิธีในภาพที่ 11 และภาพที่ 12 แสดงเอกสาร WSDL ที่ถอดรหัสแล้ว

Algorithm3: MatchDoc($Attr_{DK}$,$Attr_{EK}$)
1. $Attr_{hash}$ = H($Attr_{DK}$)
2. **If**($Attr_{hash}$ == $Attr_{EK}$)
3. Remove number from $Attr_{DK}$
4. Replace $Attr_{EK}$ of $WSDL_{EK}$ with $Attr_{DK}$
5. **Endif**

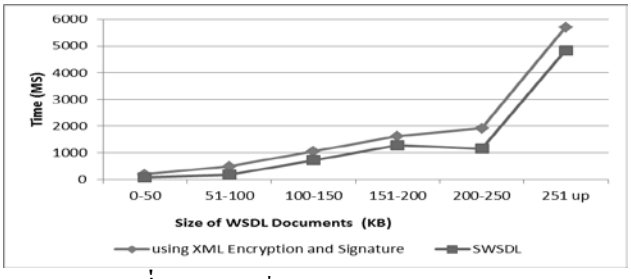
ภาพที่ 11: ขั้นตอนการจับคู่แอททริบิวต์ในเอกสาร WSDL

```
<wsdl:message name="MondkalenderInputMessage">
  <wsdl:part name="month" type="s:gMonth"/>
  <wsdl:part name="year" type="s:gYear"/>
</wsdl:message>
```

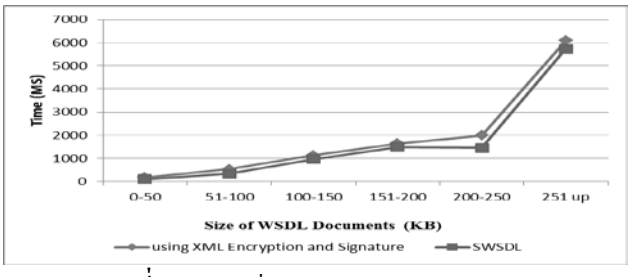
ภาพที่ 12: เอกสาร WSDL ที่ถอดรหัส

4.1 ผลลัพธ์จากการทดลอง

ทดสอบการทำงานโดยนำข้อมูลเอกสาร WSDL จาก wsdlcorpus [13] มาใช้ในการทดสอบ ซึ่งระบบได้ทดสอบกับเอกสาร WSDL1.1 โดยแบ่งช่วงขนาดของเอกสาร WSDL เป็นช่วงดังนี้คือ 6 1) ขนาด 0- 50 KB 2) ขนาด 51- 100 KB 3) ขนาด 101- 150 KB 4) ขนาด 151 - 200KB 5) ขนาด 201- 250 KB และ 6) ขนาดตั้งแต่25 1 KB ขึ้นไป ผลการทดลองในการเข้าและถอดรหัส WSDL แสดงดังภาพที่ 13 และ 14 ตามลำดับ



ภาพที่ 13 : เวลาเฉลี่ยในการเข้ารหัสเอกสาร WSDL



ภาพที่ 14: เวลาเฉลี่ยในการถอดรหัสเอกสาร WSDL

ผลการทดลองแสดงให้เห็นว่า ระยะเวลาในการเข้ารหัสและถอดรหัสเอกสาร WSDL1.1 ด้วย SWSDL ที่นำเสนอเมื่อเปรียบเทียบกับการเข้ารหัสและถอดรหัสด้วยมาตรฐานความปลอดภัยของ XML มีระยะเวลาเฉลี่ยในการเข้ารหัสและถอดรหัสเอกสาร WSDL ลดลง 37.74% และ 22.50 % เมื่อเทียบกับงานวิจัย [11] ตามลำดับ

5. บทสรุป

บทความนี้นำเสนอกลไกความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส ทำให้ผู้ใช้บริการได้รับเอกสาร WSDL ที่เป็นความลับและมีความถูกต้องสมบูรณ์จากผู้ให้บริการ และช่วยลดระยะเวลาในการเข้ารหัสและถอดรหัสข้อความในเอกสาร WSDL เมื่อเทียบกับงานวิจัย [11] โดยประยุกต์ใช้เอกสาร WSDL1.1 ร่วมกับการเข้ารหัสลับ ด้วยขั้นตอนวิธี AES RSA และกระบวนการแฮช โดยเข้ารหัสเอกสาร WSDL ในส่วนแท็ก <message> แท็ก <portType> และ แท็ก <service> เพื่อป้องกัน WSDL Scanning และ Parameter Tampering โดยงานวิจัยนี้ยังขาดส่วนของการยืนยันตัวตนของผู้ใช้บริการ ซึ่งจะพัฒนาส่วนนี้เพิ่มเติมในอนาคตเพื่อให้ได้กลไกความปลอดภัยที่มีประสิทธิภาพมากขึ้น อีกทั้งจะพัฒนาให้ครอบคลุมเอกสาร WSDL2.0 ด้วย

เอกสารอ้างอิง

[1] Lucinda Dykes and Ed Tittel, *XML for DUMMIES*, Wiley Publishing: NJ, 2005.

[2] Kiet T.Tran, *Introduction to Web Services with Java*.

[3] Manuel Mogollon, *Cryptography and Security Sevices :Mechanisms and Applications*, Cybertech Publishing: New York, 2007.

[4] Jothy Rosenberg and David L. Remy, *Securing Web Services with WS-Security*, Sams Publishing: United States of America, 2004.

[5] W3C, “World Wide Web Consortium”, [Online]. Availablefrom: http://www.w3.org [2013,September 21].

[6] Wikipedia. “Web Services Description Language”, [Online].Availablefrom:http://en.wikipedia.org/wiki/Web_Services_Description_Language [2013,September 20].

[7] WSDL Threat, “Anatomy of a Web Services attack A Guide to Threats and Preventative Countermeasures”, [Online].Availablefrom:http://csis.org/files/media/csis/insights/anatomy_of_attack_wp.pdf [2013,July 8].

[8] M.Jensen, N.Gruschka and R.Herkenhoner, “A survey of attacks on web services Classification and Countermeasures”, *Springer*, pp.185-197, 2009.

[9] N.Sidharth and J.Liu, “ IAPF:A Framework for Enhancing Web Services Security”, *Computer Software and Applications Conference*, vol.1 ,pp. 23-30, 2007.

[10] A.Mirtalebi and M.R.Khayyambashi, “Enhancing security of Web service against WSDL threats,” *2nd IEEE International Conference*, pp. 920-923, 2011.

[11] A.Mirtalebi and M.R.Khayyambashi, “A new Security Framework for Protecting WSDL File of Web Service,” *IJCSNS*, vol. 12, no.9, pp. 84-90, 2012.

[12] N.Shahgholi, M.Mohsenzadeh, M.A.Seyyedi and S.H.Qorani, “A new security framework against Web services' XML attacks in SOA,” *Next Generation Web Services Practices (NWeSP)*, pp.314-319, 2011.

[13] wsdlcorpus, “WSDL corpus”, [Online]. Availablefrom: http://coopis.sjtu.edu.cn:8080/cisg/projects/wsdlcorpus/wsdlcorpus.zip/view [2013,July 7].

ประวัติผู้เขียน

ชื่อ สกุล นางสาวเจนจิรา หวังหลี

รหัสประจำตัวนักศึกษา 5510220002

วุฒิการศึกษา

วุฒิ	ชื่อสถาบัน	ปีที่สำเร็จการศึกษา
วท.บ. (วิทยาการคอมพิวเตอร์)	มหาวิทยาลัยสงขลานครินทร์	2552

ทุนการศึกษา (ที่ได้รับในระหว่างการศึกษา)

ทุนอุดหนุนวิจัยเพื่อวิทยานิพนธ์ ประจำปีการศึกษา 2557 จากบัณฑิตวิทยาลัย มหาวิทยาลัยสงขลานครินทร์ วิทยาเขตหาดใหญ่

การตีพิมพ์เผยแพร่ผลงาน

เจนจิรา หวังหลีและลัดดา ปรีชาวีรกุล. 2557. กลไกความปลอดภัยของเอกสาร WSDL ในเว็บเซอร์วิส. การประชุมทางวิชาการระดับชาติด้านคอมพิวเตอร์และเทคโนโลยีสารสนเทศ ครั้งที่ 10 (NCCIT 2014). ภูเก็ต, ประเทศไทย, 8-9 พฤษภาคม 2557.