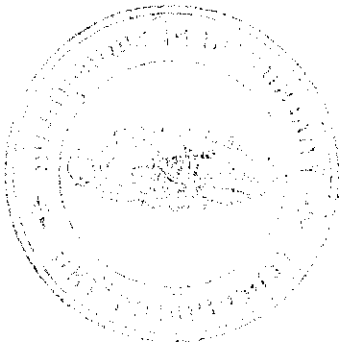


การตรวจจับการปลอมแปลงเนื้อหาอีเมล์

EMAIL SPOOFING CONTENT DETECTION

นายธณวรพงษ์ กี่ดำรงกูล รหัส 53363478

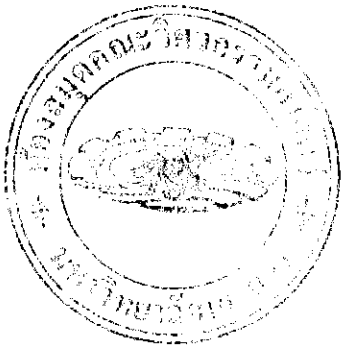
นายธีรวัต แก้วโชติ รหัส 53363508

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2556



ใบรับรองปริญญานิพนธ์

ชื่อหัวข้อโครงงาน	การตรวจจับการปลอมแปลงเนื้อหาอีเมล			
ผู้ดำเนินโครงงาน	นายธณวรพงษ์	กี่ดำรงกูล	รหัส	53363478
	นายธีรวัต	แก้วโชติ	รหัส	53363508
ที่ปรึกษาโครงงาน	อาจารย์ภาณุพงศ์ สอนคม			
สาขาวิชา	วิศวกรรมคอมพิวเตอร์			
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์			
ปีการศึกษา	2556			

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า

.....ที่ปรึกษาโครงงาน

(อาจารย์ภาณุพงศ์ สอนคม)

.....กรรมการ

(อาจารย์ ดร.สุรเดช จิตประไพกุลศาล)

.....กรรมการ

(อาจารย์เศรษฐา ตั้งค้าวานิช)

ชื่อหัวข้อโครงงาน	การตรวจจับการปลอมแปลงเนื้อหาอีเมล			
ผู้ดำเนินโครงงาน	นายธณวรพงษ์	คี่ดำรงกูล	รหัส	53363478
	นายธีรวัต	แก้วโชติ	รหัส	53363508
ที่ปรึกษาโครงงาน	อาจารย์ภาณุพงศ์ สอนคม			
สาขาวิชา	วิศวกรรมคอมพิวเตอร์			
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์			
ปีการศึกษา	2556			

บทคัดย่อ

จดหมายหลอกลวง (Spoof Email) คือการหลอกลวงทางอินเทอร์เน็ต โดยปลอมแปลงเนื้อความของอีเมล์ให้ผู้ใช้งาน (เหยื่อ) หลงเชื่อว่าเป็นอีเมล์ที่ติดต่อมาจากหน่วยงานหรือสถาบันที่ท่านทำการติดต่ออยู่ และมีการร้องขอให้ท่านกระทำการบางอย่าง เช่น ให้ทำการอัพเดต หรือยืนยันข้อมูลบัญชีของท่าน โดยทั่วไปแล้วจะมีวัตถุประสงค์ที่จะหลอกให้เหยื่อกระทำการที่อาจก่อให้เกิดความเสียหายหรือบอกข้อมูลที่มีความสำคัญออกมาเช่น รหัสผ่าน หรือข้อมูลส่วนตัว

คณะผู้จัดทำจึงได้เล็งเห็นความสำคัญของปัญหาเหล่านี้ ประกอบกับปัจจุบันมีการปลอมแปลงอีเมล์มากขึ้น ทางคณะผู้จัดทำจึงเกิดความคิดที่ว่า ทำอย่างไรถึงจะสามารถทราบว่าอีเมล์ที่เราได้รับนั้นเป็นอีเมล์หลอกลวงหรือไม่ จึงพัฒนาโครงงานนี้ขึ้นเพื่อตรวจสอบอีเมล์ปลอมแปลงโดยอัตโนมัติขึ้นมา โดยมีการนำการตรวจสอบบทความ Spam Filter มาประยุกต์ใช้ตรวจสอบผสมกับการตรวจสอบลิงค์ที่เป็นอันตราย จึงทำให้โปรแกรมมีประสิทธิภาพมากกว่าการตรวจสอบ Spam ธรรมดา จากผลการทดสอบพบว่าโปรแกรมมีอัตราของการตรวจจับอีเมล์ปลอมแปลงมีประสิทธิภาพสูงกว่า Gmail ถึง 44%

Project Title	EMAIL SPOOFING CONTENT DETECTION			
Name	Mr. Tanaworapong	Keedumrongkool	ID.	53363478
	Mr.Theerawat	Kaewchote	ID.	53363508
Project Advisor	Mr. Panupong	Sornkhom		
Major	Computer Engineering			
Department	Electrical and Computer Engineering			
Academic Year	2013			

ABSTRACT

Phishing Email (Spoof Email) is internet fraud. Forged by the body of the email to the user (victim) believe this is the contact email from the agency or institutions that you make contact. And request that you do something such as update the or confirm your account information. Typically intended to lure victims act in a manner likely to cause damage or the information that is important to come out such as passwords or personal information

The organizing committee was made aware of these issues. At present there are more fake email. The organizing has proposed an idea how to know that email is the email spoofing or not. This project was developed to check email spoofing automatically up. The leading article inspection Spam Filter applied to monitoring check blending malicious link. To make the program more effective than conventional Spam checking. The results showed that the rate of detecting email spoofing to 44% more efficient than Gmail.

กิตติกรรมประกาศ

การจัดทำโครงงานในครั้งนี้สำเร็จลุล่วงไปได้ด้วยดี คณะผู้จัดทำขอขอบพระคุณอาจารย์ที่ปรึกษาโครงงาน อาจารย์ภาณุพงศ์ สอนคม ผู้ซึ่งกรุณาสละเวลา ความคิด และประสบการณ์ ให้ความรู้ คำแนะนำ คำปรึกษาอันมีค่ายิ่ง และเอาใจใส่เป็นอย่างดีระหว่างการดำเนินโครงงาน อีกทั้งยังตรวจสอบข้อบกพร่องต่างๆ จนโครงงานนี้เสร็จสมบูรณ์

ขอขอบพระคุณอาจารย์ ดร.สุรเดช จิตประไพกุลศาล และอาจารย์เศรษฐา ตั้งค้าวานิช ที่ท่านกรุณาสละเวลามารับเป็นกรรมการตรวจสอบโครงงาน อีกทั้งยังให้คำแนะนำและตรวจสอบแก้ไขโครงงานให้สมบูรณ์ยิ่งขึ้น

ในโอกาสนี้ทางคณะผู้จัดทำโครงงานจึงขอขอบพระคุณทุก ๆ ท่านที่มีส่วนช่วยทำให้โครงงานนี้ประสบความสำเร็จลุล่วงไปได้ด้วยดี

ธณวรพงษ์ กี่ดำรงกูล
ธีรวัต แก้วโชติ

สารบัญ

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญตาราง

ตารางที่ หน้า

สารบัญรูป

รูปที่ หน้า

สารบัญรูป(ต่อ)

รูปที่ หน้า

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

จดหมายอิเล็กทรอนิกส์ (Electronic Mail) หรือเรียกสั้น ๆ ว่า อีเมล์ (E-mail) ซึ่งเป็นที่นิยมและแพร่หลายอย่างมากในปัจจุบัน อีเมล์ส่งถึงกันผ่านระบบเครือข่าย ผู้ส่งสามารถที่จะส่งอีเมล์ไปให้ผู้รับซึ่งเป็นสมาชิกของระบบอินเทอร์เน็ตได้โดยไม่จำกัดเวลาและสถานที่ อีเมล์จะส่งถึงปลายทางอย่างรวดเร็วภายในไม่กี่วินาที หรืออาจจะส่งจดหมายฉบับเดียวไปถึงผู้รับหลายคนในเวลาเดียวกันได้ อีกทั้งการส่งอีเมล์ยังสามารถแนบไฟล์ภาพ ไฟล์โปรแกรม และไฟล์ข้อมูลได้ จึงเป็นช่องทางของมิจฉาชีพในการปลอมแปลงอีเมล์ เพื่อหลอกลวงข้อมูลทางการเงิน หรือข้อมูลที่สำคัญจากผู้ใช้อีเมล์ ทำให้เกิดความเสียหายต่อผู้ใช้อีเมล์

การปลอมแปลงอีเมล์ (Email Spoofing) ซึ่งเป็นการปลอมแปลงอีเมล์แอดเดรสของผู้ส่ง กล่าวคือ ผู้ใช้จะได้รับอีเมล์ที่ระบุว่ามาจากผู้ส่งคนหนึ่ง แต่จริงๆ แล้วนั้นเป็นอีเมล์ที่มาจากผู้ส่งอีกคนหนึ่ง เนื่องจากโดยทั่วไปแล้ว การได้รับอีเมล์ สิ่งแรกที่เราจะสังเกต คือ ใครเป็นคนส่งมาให้เราถ้ามาจากคนที่เรารู้จักหรือไม่ เพื่อที่เราจะได้แน่ใจว่า ไม่เป็นสแปมเมล หรือมีไวรัสแนบมา การปลอมแปลงอีเมล์นี้โดยทั่วไปแล้วจะมีวัตถุประสงค์ที่จะหลอกให้เหยื่อกระทำการที่อาจก่อให้เกิดความเสียหายหรือบอกข้อมูลที่มีความสำคัญออกมาเช่น รหัสผ่าน หรือข้อมูลส่วนตัว

ดังนั้นทางคณะผู้จัดทำจึงได้เล็งเห็นปัญหาเหล่านี้ ประกอบกับปัจจุบันมีการปลอมแปลงอีเมล์มากขึ้น ทางคณะผู้จัดทำจึงเกิดความคิดที่ว่า ทำอย่างไรถึงจะสามารถทราบว่าอีเมล์ที่เราได้รับนั้นเป็นอีเมล์หลอกลวงหรือไม่ จึงคิดโปรแกรมในการตรวจสอบอีเมล์ขึ้นมาโดย ได้นำแนวคิดของการวิเคราะห์แบบเบย์มาประยุกต์ใช้เพื่อวิเคราะห์เนื้อหาของอีเมล์แต่ละฉบับ ร่วมกับการตรวจสอบลิงค์ที่มีในเนื้อหา

1.2 วัตถุประสงค์ของโครงงาน

1.เพื่อพัฒนาระบบที่ตรวจสอบอีเมล์ ว่าเป็นอีเมล์ที่ปลอมแปลงมาหรือไม่

1.3 ขอบเขตของโครงงาน

1. รองรับการทำงานของ Gmail Account บน Host ของ Gmail
2. ทำการตรวจสอบเฉพาะเนื้อหาของอีเมล์
3. ตรวจสอบอีเมล์โดยแสดงผลเป็นการตรวจสอบ 3 รูปแบบคือ อีเมล์ปลอมแปลง อีเมล์น่าสงสัย หรืออีเมล์ปลอดภัย
4. ใช้ TomCat 7.0 ในการรันเซิร์ฟเวอร์

1.4 ขั้นตอนการดำเนินการ

ตารางที่ 1.1 ตารางขั้นตอนและแผนการดำเนินโครงงาน

กิจกรรม	ปี 2556								
	เม.ย	พ.ค.	มิ.ย.	ก.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.
1.ศึกษาค้นคว้าข้อมูลในการทำโครงงาน	←	→							
2.ศึกษาโครงสร้างภาษา Java Servlet		←	→						
3.ศึกษาข้อมูลเกี่ยวกับอัลกอริทึมที่ช่วยในการตรวจสอบข้อความภายในอีเมล์			←	→					
4.ศึกษาข้อมูลเกี่ยวกับอัลกอริทึมที่ช่วยในการตรวจสอบลิงก์				↔					
5.ทำการออกแบบโปรแกรมและกำหนดส่วนประกอบ					←	→			
6.ทดสอบการใช้งานและแก้ไขข้อบกพร่อง						←	→	→	
7. สรุปผลการทำโครงงานและจัดทำรายงาน									↔

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. เพื่อให้ผู้ที่ใช้โปรแกรมทราบถึงความปลอดภัยหรือความเสี่ยงของอีเมล์นั้น
2. เพื่อช่วยลดปัญหาเรื่องการถูกหลอกลวงจากการปลอมแปลงอีเมล์

1.6 งบประมาณที่ต้องการใช้ในการดำเนินงาน

1. ค่าใช้จ่ายเกี่ยวกับเอกสาร	1,000	บาท
2. ค่าใช้จ่ายอื่นๆ	1,000	บาท
รวม	**2,000**	**บาท**

หมายเหตุ – ถัวเฉลี่ยทุกรายการ

บทที่ 2

หลักการและทฤษฎี

2.1 หลักการและทฤษฎีเบื้องต้น

2.1.1 E-mail [11]

อีเมล์หรือจดหมายอิเลคทรอนิกส์ เป็นการติดต่อสื่อสารกันด้วยวิธีที่ใช้รับส่งกันโดยผ่านเครือข่ายคอมพิวเตอร์แลกเปลี่ยนข่าวสารระหว่างเครื่องคอมพิวเตอร์ โดยผ่านระบบโทรคมนาคม ข่าวสารหรือข้อความของอีเมล์จะเป็นไฟล์ประเภทข้อความ อย่างไรก็ตามสามารถส่งไฟล์ประเภทอื่น เช่น ไฟล์ประเภทภาพหรือเสียง เป็นไฟล์ที่แนบไปในรหัสแบบ binary อีเมล์สามารถแลกเปลี่ยนระหว่างผู้ใช้ของ online service provider กับระบบเครือข่ายอื่น ผู้ใช้งานเพียงแค่ทราบบัญชีอีเมล์ (E-mail Account) ของผู้รับ ก็สามารถที่จะส่งข้อความไปยังผู้รับได้ การสื่อสารประเภทนี้ได้รับความนิยมและถูกนำมาใช้งานอย่างแพร่หลายในปัจจุบัน เนื่องจากความสะดวกในการใช้บริการ กล่าวคือ ไม่ว่าใช้บริการจะอยู่ที่ไหนก็ตามหากผู้ใช้งานเชื่อมต่อเข้ากับเครือข่ายอินเทอร์เน็ตก็สามารถที่จะรับ หรือส่งอีเมล์ได้ตลอดเวลา

2.1.2 Spoof Email [1]

จดหมายหลอกลวง (Spoof Email) คือการหลอกลวงทางอินเทอร์เน็ต โดยปลอมแปลงเข้ามาในรูปแบบของอีเมล์ (Email Spoofing) โดยปลอมแปลงเนื้อความของอีเมล์ให้ผู้ใช้งาน (เหยื่อ) หลงเชื่อว่าเป็นอีเมล์ที่ติดต่อมาจากหน่วยงานหรือสถาบันที่ท่านทำการติดต่ออยู่ และมีการร้องขอให้ท่านกระทำการบางอย่าง เช่น ให้ทำการอัพเดต หรือยืนยันข้อมูลบัญชีของท่าน ซึ่งหากท่านไม่ตอบกลับอีเมล์ดังกล่าว อาจก่อให้ท่านไม่สามารถใช้งานได้ นอกจากนี้เพื่อให้อีเมล์ปลอมนั้นสมจริงมากขึ้น จะมีการใส่ hyperlink รวมในเนื้อความของอีเมล์ เพื่อให้เหมือนกับว่าเป็นการส่งจาก URL ของหน่วยงานหรือสถาบันนั้นจริงๆ ซึ่งโดยแท้จริงแล้ว เป็นเว็บไซต์ปลอมที่สร้างขึ้นมา หรือที่เรียกว่า “Phishing/Spoofed Website” เมื่อท่านหลงเชื่อเข้าไปในเว็บไซต์หลอกลวงนี้ และกรอกข้อมูลที่เป็นความลับส่วนตัว เช่น ข้อมูลทางการเงินหรือข้อมูลส่วนตัวต่างๆ เช่น ข้อมูลหมายเลขบัตรเครดิต หมายเลขประจำตัวผู้ใช้ (Username) รหัสผ่าน (Password) หมายเลขประจำตัว เว็บไซต์

หลอกลวงนี้จะนำข้อมูลของท่านไปใช้ประโยชนในทางที่ผิด โดยกระทำการในนามของท่านซึ่งทำให้เกิดความเสียหายแก่ท่านได้

2.1.3 วิธีการ Bayesian ของ Graham [2]

Graham [2] ได้นำแนวคิดของการวิเคราะห์แบบเบย์มาประยุกต์ใช้เพื่อวิเคราะห์เนื้อหาของอีเมล์แต่ละฉบับ โดยที่ข้อมูลที่ใช้วิเคราะห์คือ เนื้อหาของอีเมล์ สิ่งที่ได้จากการวิเคราะห์คือ ค่าความน่าจะเป็น ซึ่งสามารถนำมาใช้ในการตัดสินว่า อีเมล์ฉบับนั้นเป็นอีเมล์ที่ดี (HAM) หรือเป็นอีเมล์ปลอมแปลงได้ โดยเมื่อนำทฤษฎีของเบย์ไปประยุกต์ใช้ในการสร้างตัวกรอง อีเมล์ปลอมแปลงจะได้ ด้วยนักพัฒนาโปรแกรมนามว่า Rennie Jason ในปี 1996โปรแกรมใช้หลักความน่าจะเป็นแบบ Bayes ในการจัดการ โดยพิจารณาจากคำแต่ละคำในข้อความนั้น เช่นถ้าเรากำลังพิจารณาคำว่า "Viagra" ใช้สมการคำนวณเป็น สมการ [3] ดังนี้

$$\Pr(S|W) = \frac{\Pr(W|S) \cdot \Pr(S)}{\Pr(W|S) \cdot \Pr(S) + \Pr(W|H) \cdot \Pr(H)} \qquad [3]$$

Pr(S|W) คือความน่าจะเป็นที่ ข้อความนั้นจะเป็น ข้อความปลอมแปลง โดยรู้จากคำว่า "Viagra"

Pr(S) คือความน่าจะเป็นที่จะให้ข้อความใดๆ เป็นข้อความปลอมแปลง

Pr(W|S) คือความน่าจะเป็นที่ ข้อความว่า "Viagra" จะเป็นข้อความปลอมแปลง

Pr(H) คือความน่าจะเป็น ที่ข้อความใดจะไม่เป็นข้อความปลอมแปลง

Pr(W|H) คือความน่าจะเป็น ที่คำว่า "Viagra" จะอยู่ในข้อความที่ไม่ใช่ข้อความปลอมแปลง

โดยตัวกรองอีเมล์ปลอมแปลง แบบเบย์ของ Graham นี้จะทำการ สร้างรายการของทุกคำที่ปรากฏอยู่ในข้อความอีเมล์ จากนั้นโปรแกรมจะพิจารณาว่าข้อความนี้ ผ่านการพิจารณาหรือไม่ โดยที่ตัวกรองนี้จะเพิ่มรายการคำนั้นเข้าไว้ในรายการแยกประเภท จัดไว้เป็นอีเมล์ดีหรือเป็น อีเมล์ปลอมแปลง ด้วยวิธีการเรียนรู้แยกแยะ คำไหนดี คำไหนไม่ดี ทำให้สามารถปรับตัวเข้ากับ อีเมล์

ปลอมแปลง และอีเมล์ใหม่ๆ โดยที่มีความถูกต้องยอมรับได้ เมื่อใดที่เกิดมีการทำงานผิดพลาดก็สามารถปรับแก้ได้

ข้อดีของ Bayesian คือ

ในการวิเคราะห์ข้อมูลด้วยวิธีการแบบเบย์ มีความสามารถที่จะวิเคราะห์ข้อมูลอีเมล์ที่รับเข้ามาได้โดยที่ทำการพิจารณาจากเนื้อหาของอีเมล์ซึ่งเป็นวิธีการเดียวกับที่มนุษย์ใช้ด้วย เหตุเองนี้จึงทำให้ผลการวิเคราะห์ด้วยวิธีการนี้มีค่าความถูกต้องสูง

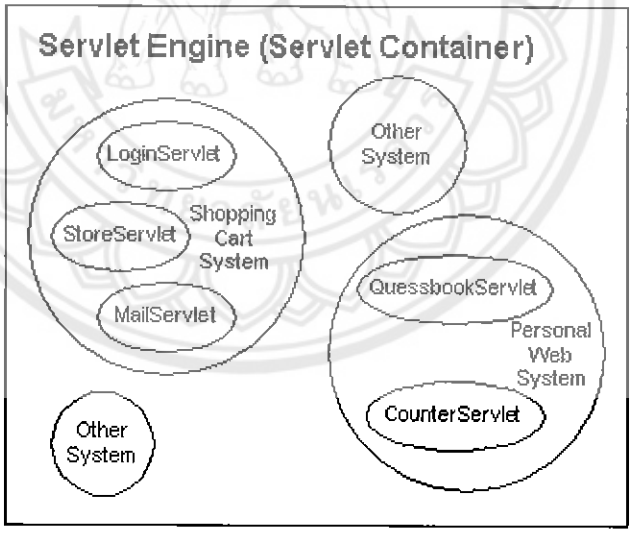
ข้อจำกัดของ Bayesian คือ

ในการวิเคราะห์ข้อมูลด้วยวิธีแบบเบย์ นั้นมีความจำเป็นที่จะต้องใช้เวลาในการประมวลผลและทรัพยากรของเครื่องคอมพิวเตอร์เป็นอย่างมาก เพราะการวิเคราะห์ด้วยวิธีการนี้จะต้องทำการวิเคราะห์ทุกข้อความในอีเมล์ ที่รับเข้ามาซึ่งจำเป็นต้องใช้การประมวลผลที่มากกว่าวิธีการแบบอื่น

2.1.4 Java Servlet [4]

Servlet เป็น Server Side Application แบบหนึ่งซึ่งอ้างอิงคอนเซ็ปมาจาก CGI ข้อดีของ Servlet ที่อยู่เหนือ CGI อย่างแรกก็คือตัวภาษาที่ใช้เขียนซึ่งก็คือจาว่านั่นเอง จาวาเป็นภาษาที่ใช้คอนเซ็ปของ Object Oriented ในการเขียน หลายคนที่เกี่ยวข้องกับการเขียนโปรแกรมสำหรับโปรเจคใหญ่ ๆ จะทราบดีว่า Object Oriented สามารถลดความซับซ้อนของโครงสร้างโปรแกรมรวมไปถึงการอำนวยความสะดวกในการ reuse ส่วนของโปรแกรมที่เขียนไว้แล้วเพียงไร นอกจากนี้จาว่ายังเป็นภาษาที่เป็นลักษณะแบบ platform independent ซึ่งจะช่วยให้เราสามารถที่จะทำการพัฒนาระบบโดยใช้ Environment อะไรก็ได้ซึ่งโดยทั่วไปมักนิยมใช้ Window Environment โดยจะนำโปรแกรมที่เขียนเสร็จแล้วมารันบน Unix Environment เพื่อเพิ่มความเสถียรภาพของโปรแกรมแทน นอกจากนี้ Servlet ยังมีความเร็วที่สูงกว่า CGI เพราะ Servlet ใช้หลักการของ thread โดยจะทำการสร้าง 1 thread ต่อหนึ่ง request ที่มาจาก client ซึ่งในทางกลับกัน CGI จะทำการสร้าง 1 process ต่อหนึ่ง request* ซึ่งจะทำให้เปลืองทรัพยากรมากกว่าและ process ในการรันก็จะช้ากว่าด้วย ท้ายที่สุดจุดเด่นที่สำคัญของ Servlet ก็คือ API (Application Programming Interface) โดยระบบที่

ทำการพัฒนาโดยใช้คอนเซ็ปของ Servlet จะสามารถเรียกใช้ API ที่ทางจาว่ามีมาให้ (javax.servlet.*, javax.servlet.http.*) ซึ่งจะช่วยทำให้การพัฒนาระบบดังกล่าวง่ายและเร็วยิ่งขึ้น Servlet Engine ในการรันระบบที่เขียนขึ้นโดยใช้หลักการของ Servlet เราจะต้องนำระบบดังกล่าวมาบรรจุอยู่ในสิ่ง ๆ หนึ่งที่เรียกว่า Servlet Engine ให้นึกถึง Servlet Engine คล้าย ๆ กับกล่อง ๆ หนึ่งที่ใส่ลูกปิงปองไว้หลายลูก โดยลูกปิงปองแต่ละลูกก็คือระบบ ๆ หนึ่งนั่นเอง หลายคนอาจสงสัยทำไมถึงใช้คำว่าระบบ โดยทั่วไป Server Side Application หนึ่ง ๆ ที่ถูกเขียนขึ้นโดยใช้ Servlet API จะถูกเรียกว่า Servlet ในหนึ่งระบบอาจประกอบด้วย Servlet หลายอัน ยกตัวอย่างเช่น ระบบที่เกี่ยวกับ Shopping Cart อาจจะประกอบด้วย Servlet ที่ทำหน้าที่ในการเช็คล็อกอิน, Servlet ที่ทำหน้าที่ในการเก็บข้อมูลสินค้า, Servlet ที่ทำหน้าที่ในการส่งเมล์กลับไปยังลูกค้าเพื่อบอกว่าได้ทำการส่งของไปให้แล้ว เป็นต้น ดังนั้นถ้ามองโดยรวมแล้ว Servlet Engine ก็คือที่รวมของระบบตั้งแต่หนึ่งระบบถึงหลายระบบ โดยแต่ละระบบจะประกอบด้วย Servlet หนึ่งอันหรือมากกว่าระบบ ในที่นี้อาจจะหมายถึง Zone (Apache JServ) หรือ Web Application (Tomcat) ก็ได้ดังรูปที่ 2.5



รูปที่ 2.1 Servlet Engine and its Servlets [4]

Servlet Engine เป็นเพียงกล่อง ๆ หนึ่งที่ใช้บรรจุและรันกลุ่มของ Servlet เท่านั้น ในการที่จะทำการติดต่อสื่อสารกับ Client ตัว Servlet Engine นี้จะต้องทำงานร่วมกัน Web Server ซึ่งเปรียบเสมือนฉากหน้าที่ติดต่อกับ Client อีกทีหนึ่ง เมื่อใดก็ตามที่มี request ส่งมาจาก Client ถ้า

request นั้นเจาะจงมาที่ตัว Servlet ทาง Web Server ก็จะทำการ forward ตัว request นั้นมาให้ Servlet Engine ซึ่งทาง Servlet Engine ก็จะทำการเรียก Servlet ที่ Client ต้องการขึ้นมาทำการประมวลผล request นั้น โดยท้ายสุด Servlet จะส่งผลกลับไปให้ Servlet Engine, Servlet Engine ก็จะ forward ผลที่ได้กลับไปให้ Web Server ซึ่ง Web Server ก็จะส่งผลกลับไปให้ Client Servlet Engine อาจจะเป็นส่วนที่ติดมากับ Web Server อยู่แล้วยกตัวอย่างเช่น Servlet Engine ที่อยู่ใน Netscape Enterprise Server, IBM WebSphere หรืออาจจะเป็นส่วนที่เป็น Add-on ให้กับ Web Server ก็ได้เช่น Apache Jserv, Tomcat, JRun หรือแม้กระทั่งเป็นส่วนหนึ่งที่อยู่ใน Application Server เช่น BEA Weblogic เป็นต้น ทั้งนี้การเลือกใช้ Servlet Engine แต่ละชนิดก็มักขึ้นอยู่กับปัจจัยหลายอย่างเช่น ความสะดวกในการรวมระบบที่จะสร้างขึ้นมาใหม่กับระบบที่มีอยู่แล้ว, งบประมาณที่มีอยู่สำหรับโครงการหรืออาจจะรวมไปถึงทักษะและประสบการณ์ส่วนตัวของนักพัฒนาแต่ละคน

2.1.5 IMAP [5]

Internet Message Access Protocol (IMAP) เป็นมาตรฐานโปรโตคอล สำหรับการเข้าถึง e-mail จากเครื่อง local service โดย IMAP เป็นโปรโตคอลแบบ client/service ซึ่ง e-mail จะได้รับและเก็บไว้ในเครื่องแม่ข่ายอินเตอร์เน็ต ผู้ใช้สามารถดูหัวข้อ และผู้ส่งของจดหมายแล้ว จึงตัดสินใจดาวน์โหลด ผู้ใช้สามารสร้างและควบคุมโฟลเดอร์ หรือ mail box บนเครื่องแม่ข่าย ลบจดหมายหรือค้นหา IMAP ต้องการเข้าถึงแม่ข่ายอย่างต่อเนื่องตลอดช่วงเวลาการใช้ e-mail โปรโตคอล ที่มีความซับซ้อนน้อยกว่า คือ Post Office Protocol 3 (POP 3) การใช้ POP 3 ทำให้ e-mail ของผู้ใช้ได้รับการเก็บไว้ใน mail box บนเครือข่าย เมื่อต้องการอ่าน e-mail สามารถทำการดาวน์โหลด มายังคอมพิวเตอร์ของผู้ใช้ และไม่จำเป็นต้องเก็บไว้บนแม่ข่าย IMAP สามารพิจารณาเป็น remote file server ส่วน POP สามารถพิจารณาเป็นการบริการแบบ "เก็บ และ ส่ง " POP และ IMAP เกี่ยวข้องกับการรับ e-mail ของผู้ใช้ในเครื่อง local server และอย่าสับสนกับ Simple Mail Transfer Protocol (SMTP) ซึ่งเป็นโปรโตคอลสำหรับการส่ง e-mail ระหว่างจุดบนอินเตอร์เน็ต การส่ง e-mail ใช้ SMTP การอ่าน e-mail ใช้ POP และ IMAP

ข้อเสียของโปรโตคอล IMAP [6]

โปรโตคอลมีความซับซ้อน และยากในการ Implement มีซอฟแวร์ที่สนับสนุนน้อยกว่า POP IMAP เหนือกว่า POP ใน 3 ส่วนหลักๆ คือ มีคำสั่งในการจัดการตู้จดหมายจำนวนมาก มีความสามารถในการจัดการ folder อื่นๆ นอกเหนือจาก inbox มีจุดเด่นในการเพิ่มประสิทธิภาพของแบบ online โดยเฉพาะกับจดหมายที่เป็น MIME และเพราะว่าขณะนี้มีการแจก development libraries ของ IMAP ฟรี ดังนั้นความซับซ้อนของมันคงไม่มีผลต่อความนิยมใช้ที่จะเพิ่มมากขึ้นในอนาคต โดยเน็ตสเคปวางแผนที่จะรวม IMAP เข้าไว้ในเมล์เซิร์ฟเวอร์รุ่นต่อไปของตน ซึ่งน่าจะออกมาได้ในปีนี้ ยิ่งไปกว่านั้น SunSoft ก็มี IMAP Server และ Client ขณะที่ยังมี IMAP Client ที่ชื่อว่า Embla ของ ICL และ ICL/TeamWare ที่ให้ Internet Messaging Server ที่สนับสนุน POP และ IMAP ส่วนผลิตภัณฑ์อื่นๆ ที่ร่วมขบวนของ IMAP ก็ได้แก่ Control Data Mail Hub server,NetManage Z-Mail Pro และ messaging server ที่มาจาก Software.com

2.1.6 Apache [7]

Apache คืออะไร Apache คือ Software ที่ทำหน้าที่เป็น webserver โดยให้บริการ protocol HTTP ที่ port 80 ลักษณะเด่น คือเป็น Software ที่เป็น Opensource ติดตั้งมาพร้อมกับ ระบบปฏิบัติการ Linux และมีใช้กันอย่างแพร่หลายมากที่สุดในโลก ที่มาของชื่อ Apache มาจากกลุ่มคนที่ช่วยสร้างแพตช์ไฟล์สำหรับโครงการ NCSA httpd1.3 ซึ่งกลายมาเป็นที่มาของชื่อ A PAtCHy server และในอีกความหมายหนึ่งยังกล่าวถึงเผ่าอะแพชีหรืออาปาเช่ ซึ่งเป็นเผ่าอินเดียนแดงที่มีความสามารถในการรบสูงประวัติของ Apache พัฒนามาจาก HTTPD Web Server ที่มีกลุ่มผู้พัฒนาอยู่ก่อนแล้ว โดย ร็อบ แม็คคูล (Rob McCool) ที่ NCSA (National Center for Supercomputing Applications) มหาวิทยาลัยอิลลินอยสมหาวิทยาลัยอิลลินอยส์ เออร์แบนา-แชมเปญจน์ สหรัฐอเมริกา แต่หลังจากที่ แม็คคูล ออกจาก NCS และหันไปให้ความสนใจกับโครงการอื่นๆ มากกว่าทำให้ HTTPD เว็บเซิร์ฟเวอร์ ถูกปล่อยทิ้งไม่มีผู้พัฒนาต่อ แต่เนื่องจากเป็นซอร์ฟแวร์ที่อยู่ภายใต้ลิขสิทธิ์ กนู คือ ทุกคนมีสิทธิ์ที่จะนำเอาซอร์สโค้ดไปพัฒนาต่อได้ ทำให้มีผู้ใช้กลุ่มหนึ่งได้พัฒนาโปรแกรมขึ้นมาเพื่ออุดช่องโหว่ ที่มีอยู่เดิม (หรือ แพช) และยังได้รวบรวมเอาข้อมูลการพัฒนา และการแก้ไขต่างๆ แต่ข้อมูลเหล่านี้อยู่ตามที่ต่างๆ ไม่ได้รวมอยู่ในที่ที่เดียวกัน จนในที่สุด

ไบอัน บีเลนดอร์ฟ (Brian Behlendorf) ได้สร้างจดหมายกลุ่ม (mailing list) ขึ้นมาเพื่อนำเอาข้อมูลเหล่านี้เข้าไว้เป็นกลุ่มเดียวกัน เพื่อให้สามารถเข้าถึงข้อมูลเหล่านี้ได้ง่ายยิ่งขึ้น และในที่สุด กลุ่มผู้พัฒนาได้เรียกตัวเองว่า กลุ่มอาปาเช่ (Apache Group) และได้ปล่อยซอฟต์แวร์ HTTPD เว็บเซิร์ฟเวอร์ ที่พัฒนาโดยการนำเอาแพชหลายๆ ตัวที่ผู้ใช้ได้พัฒนาขึ้นเพื่อปรับปรุงการทำงาน ของซอฟต์แวร์ตัวเดิมให้มีประสิทธิภาพมากยิ่งขึ้นตั้งแต่ ปี พ.ศ. 2539 Apache ได้รับความนิยมขึ้นเรื่อยๆ จนปัจจุบันได้รับความนิยมเป็นอันดับหนึ่ง มีผู้ใช้งาน อยู่ประมาณ 65% ของเว็บเซิร์ฟเวอร์ที่ให้บริการอยู่ทั้งหมด

โดยหน้าที่หลักของ webserver ทั่วไปมีดังนี้

คอยจัดการ Request ก็คือ การร้องขอข้อมูล

คอยจัดการ Response ก็คือ การส่งข้อมูลกลับไป

คอยจัดการ process และจัดลำดับ ของ request และ response

คอยเก็บ logs ที่มีการ access เข้ามารวมกระทั้ง error ต่างๆๆ ที่ webserver พบ เช่น ไม่เจอไฟล์ชื่อนี้

สามารถ เอา module มาใช่ร่วมกับ webserver ได้ ยกตัวอย่างเช่น Apache นั้น ไม่สามารถ run ไฟล์ .php ได้ ต้อง เรียกใช้ module php อีกที หรือ การ rewrite url ก็ใช่เดียวกันต้องใช้ mod_rewrite ในการสร้าง

2.1.7 Apache Tomcat [8]

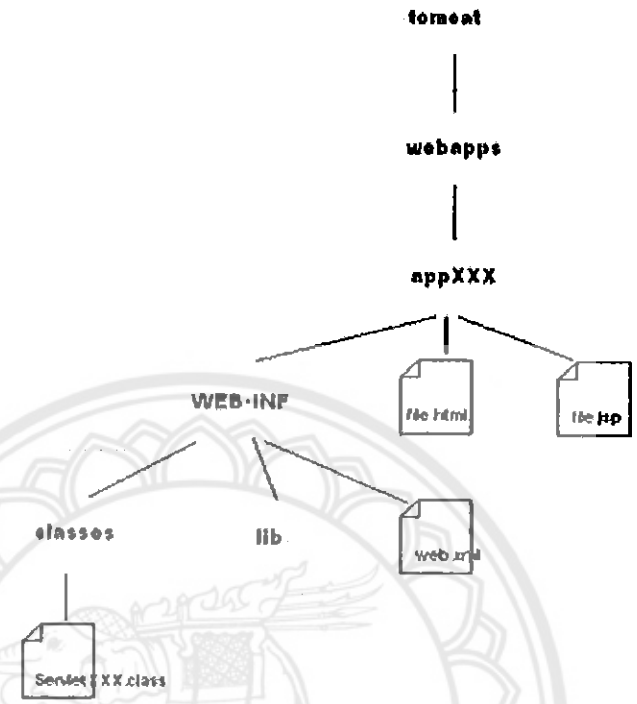
Apache Tomcat เป็นโปรแกรมบรรจุเว็บ (web container) ที่พัฒนาโดยมูลนิธิซอฟต์แวร์อะแพชี ทอมแคตใช้ข้อกำหนดของเซิร์ฟเลตและเจเอสพีจากซันไมโครซิสเต็มส์มาเป็นต้นแบบในการทำงาน ซึ่งกำหนดสภาพแวดล้อมสำหรับโค้ดจาวาเพื่อทำงานบนเว็บเซิร์ฟเวอร์ นอกจากนั้นทอมแคตได้เพิ่มเครื่องมือสำหรับการจัดการการตั้งค่าที่เก็บในรูปแบบแฟ้มเอกซ์เอ็มแอล และมีโปรแกรม HTTP เซิร์ฟเวอร์อยู่ในตัวเอง อะแพชี ทอมแคต เคยเป็นโครงการย่อยของโครงการจาการ์ตา แต่ปัจจุบันได้แยกตัวออกมาเป็นโครงการหลักของมูลนิธิซอฟต์แวร์อะแพชีทอมแคตเป็น

เว็บเซิร์ฟเวอร์ที่รองรับเซิร์ฟเลตและเจเอสพี โดยทำงานร่วมกับตัวแปลโปรแกรมชื่อ ทอมแคต แจสเปอร์ (Tomcat Jasper) ในการแปลงเจเอสพีให้กลายเป็นเซิร์ฟเลตก่อนนำไปประมวลผลเอนจินเซิร์ฟเลตของทอมแคตนั้นมักทำงานร่วมกับ อะแพชี เว็บเซิร์ฟเวอร์ (Apache HTTP Server) หรือโปรแกรมเว็บเซิร์ฟเวอร์อื่นๆ หรือสามารถตั้งตัวเป็นเซิร์ฟเวอร์เอกเทศก็ได้ ซึ่งในการพัฒนาก่อนหน้านี้มีแนวความคิดว่า เมื่อทอมแคตทำงานเป็นโปรแกรมที่ทำงานโดดเดี่ยว (standalone) จะเหมาะกับสภาพแวดล้อมที่ไม่ต้องการความรวดเร็วและการดูแลธุรกรรม (transaction) มากนัก อย่างไรก็ตามแนวความคิดดังกล่าวไม่มีอีกต่อไป เนื่องจากทอมแคตได้เพิ่มประสิทธิภาพเป็นเซิร์ฟเวอร์ที่รองรับสภาพแวดล้อมที่มีการจราจรหนาแน่นสูง ทอมแคตสามารถทำงานได้ข้ามระบบปฏิบัติการ เพียงแค่ต้องการจาวารันไทม์เอนไวรอนเมนต์ (Java Runtime Environment) เท่านั้นสมาชิกของมูลนิธิซอฟต์แวร์อะแพชี และอาสาสมัครอิสระจะเป็นผู้ช่วยพัฒนาและดูแลรักษาทอมแคต ผู้ใช้ทั่วไปสามารถเข้าถึงทั้งซอร์สโค้ดและซอฟต์แวร์ที่แปลแล้วของทอมแคตภายใต้สัญญาอนุญาตอะแพชี(Apache License) รุ่นแรกของทอมแคตที่เผยแพร่สู่สาธารณะเริ่มต้นที่ 3.0.x (ซึ่งรุ่นก่อนหน้านี้เป็นการเผยแพร่ภายในชันไมโครซิสเต็มส์เท่านั้น ไม่ออกสู่สาธารณะ) และรุ่น 6.0.13 เป็นรุ่นล่าสุดที่เสถียรในสายรุ่น 6.0.x ตามข้อกำหนดเซิร์ฟเลตรุ่น 2.5 ในปี ค.ศ. 2007

Web Container [8] มีไว้สำหรับรับ request เพื่อส่งต่อให้กับ servlet แต่ละตัว และยังช่วยการทำงานด้านอื่นๆ เพื่อให้การเขียน servlet ทำได้ง่ายขึ้น และผู้พัฒนาสามารถมุ่งไปที่ business ที่ต้องการจัดการมากกว่าจะต้องมาจัดการกับสิ่งรอบข้าง

ส่วนที่ Web container จัดการให้มีดังนี้

- Communication support ช่วยจัดการเรื่อง socket และ network ต่างๆ
- Lifecycle management จัดการ life cycle ของ servlet
- Multithreading support จัดการการทำงานของ thread เนื่องจากแต่ละ request หมายถึงการสร้าง thread 1 ตัวขึ้นมาเพื่อรับ request
- Declarative security จัดการเรื่อง security ซึ่งสามารถกำหนดได้จาก deployment descriptor (web.xml)
- JSP support



รูปที่ 2.2 แสดงถึงโครงสร้างเว็บใน Tomcat [8]

แต่อย่างไรก็ตามสำหรับการทำงานของ Servlet สามารถใช้แค่ web container ได้ ไม่จำเป็นต้องใช้ application server

Tomcat Version	Servlet	JSP Version	JSTL Version
6	2.5	2.1	1.2
5	2.4	2.0	1.2

ตารางที่ 2.2.3.1 แสดงถึงการ implement JSP และ JSTL ใน Tomcat [9]

สำหรับการ config อีกเรื่องหนึ่งของ Tomcat ก็คือ Authentication ซึ่ง Tomcat ใช้ tomcat-users.xml ที่อยู่ใน conf/ directory ในการกำหนดผู้ใช้ ตัวอย่างเช่น Tomcat manager ที่เป็น web application ที่มากับ tomcat ก็จะใช้ตัวนี้ในการ Authenticate ผู้ใช้ ซึ่งผู้ที่จะใช้ web application นี้ได้จะต้องมี role เป็น manager ดังนี้

```
<tomcat-users>
<role rolename="manager"/>
<user username="admin" password="xxxx" roles="manager"/>
</tomcat-users>
```

บทที่ 3

วิธีการดำเนินโครงงานวิศวกรรม

ในบทนี้จะกล่าวถึงขั้นตอนและวิธีการดำเดินการเพื่อให้ได้โปรแกรม การตรวจจับการปลอมแปลงอีเมล ซึ่งประกอบไปด้วย การศึกษารวบ รวมข้อมูล การออกแบบและการใช้งานของโปรแกรม โดยมีรายละเอียดดังนี้

3.1 ศึกษาการทำงาน

3.1.1 การเขียนโปรแกรม Java Servlet

ศึกษาวิธีการเขียนโปรแกรมเว็บแอปพลิเคชันด้วย Java Servlet ที่จะใช้ในการ Login เข้าเมลของ Gmail ผ่านทาง IMAP แล้วนำเมลที่ได้นั้นมาทำการตรวจสอบผ่านระบบ คัดกรองข้อความ และประมวลผล เพื่อนำมาแสดงบนหน้าเว็บไซต์

3.1.2 Spam Filtering using Bayesian Analysis

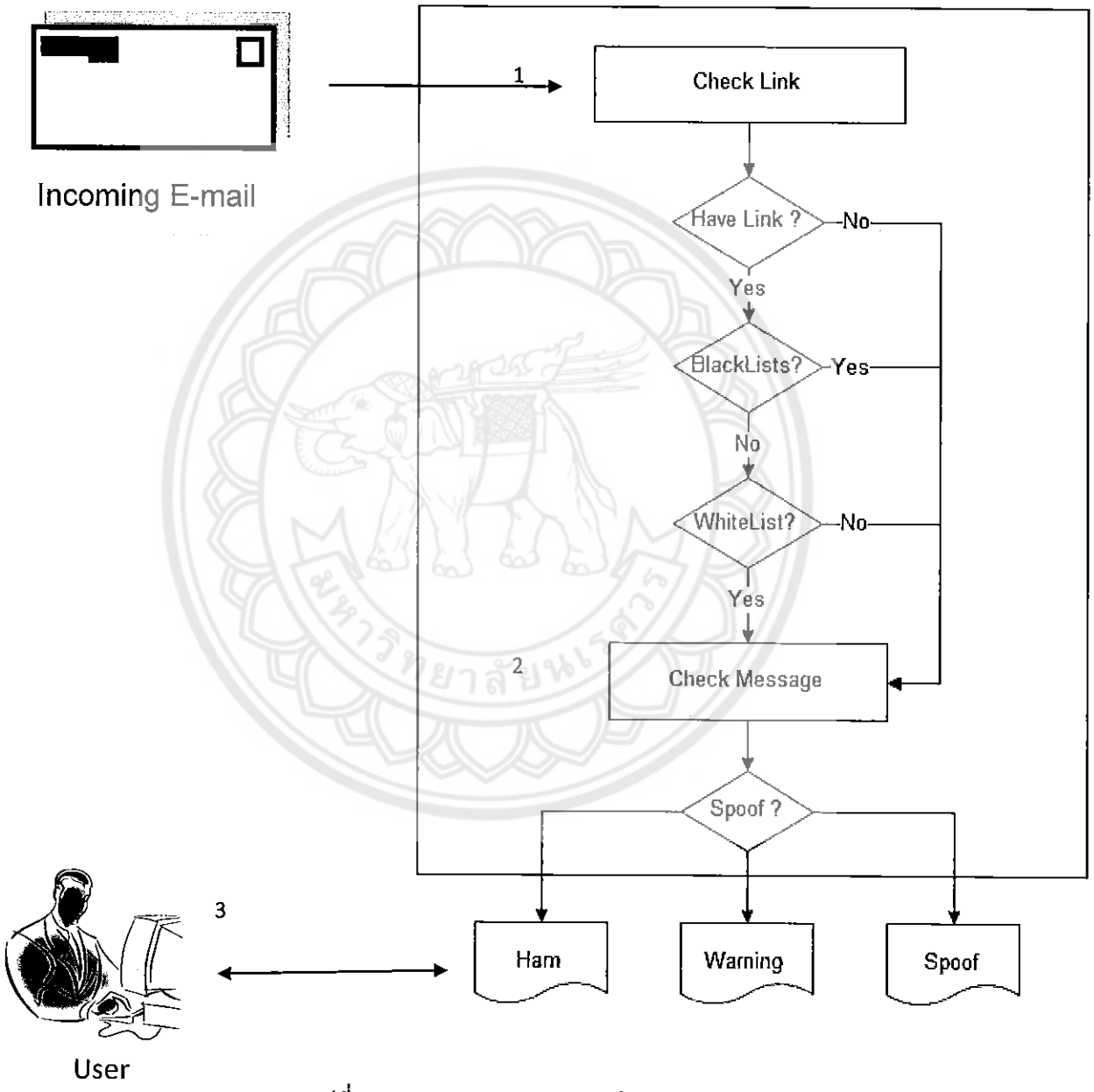
เป็นโปรแกรมใช้หลักความน่าจะเป็นแบบ Bayes ในการจัดการ โดยพิจารณาจากคำแต่ละคำในข้อความนั้น โดยนำการคัดกรองเนื้อหาสแปมเพื่อนำมาประยุกต์ใช้กับ การคัดกรองเนื้อหาของอีเมลปลอมแปลง

3.1.2 IMAP

มีความสามารถในการเข้าถึงทั้งแบบ offline และแบบ online โดยในแบบ online จดหมายจะไม่ถูกดึงมา แต่จะเป็นแบบโต้-ตอบกับ server นั่นคือผู้ใช้สามารถดึงเฉพาะหัวข้อจดหมาย , บางส่วนของจดหมาย หรือค้นหาจดหมายที่ตรงความต้องการ เพื่อนำมาทำการวิเคราะห์คัดกรองเนื้อหา

3.2 การออกแบบ

ผู้จัดทำได้มีแนวทางในการออกแบบโปรแกรมการตรวจจับการปลอมแปลงอีเมล ดังรูปที่ 3.1



รูปที่ 3.1 Diagram การทำงานของโปรแกรม

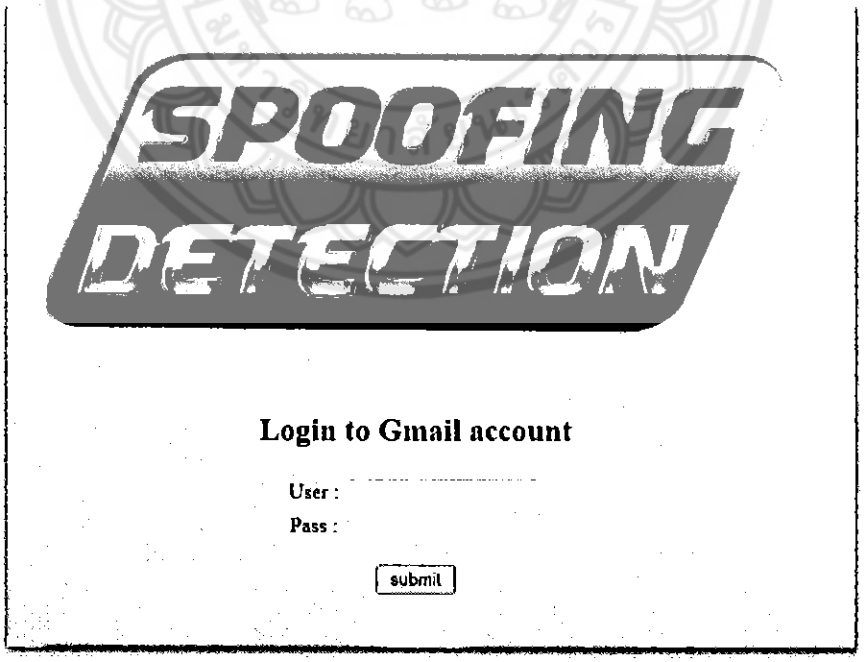
3.2.1 การออกแบบ Java Servlet

3.2.1.1 ออกแบบหน้า Login

ใช้การเขียน HTML ในการออกแบบหน้า Login โดยมี form เป็นตัวเชื่อมไปยังโปรแกรม

```
16    <div id="login" >
17
18          <div id="loginwelcome">
19            <p><h2>Login to Gmail account</h2>
20          </div>
21    <form action="ListMail" method="POST">
22    <p>
23        <table>
24        <tr>
25        <td>User : </td><td><input name="user" /></td>
26        </tr>
27        <tr>
28        <td>Pass : </td><td><input type="password" name="pass" /></td>
29        </tr>
30        </table>
31
32        <p><input type="submit" value="submit" />
33
34
35    </form>
```

รูปที่ 3.2 เขียนหน้า Login โดยใช้ HTML



รูปที่ 3.3 แสดงหน้า Login

3.2.1.2 หน้า Mail Inbox

เมื่อ Login แล้ว จะมีการส่งค่า Username และ Password มาที่ส่วนของ Inbox เพื่อที่จะทำการ เปิด IMAP จากนั้นทำการดึงค่าต่างๆของอีเมลมาแสดงผล

```
public class ListMail extends HttpServlet {
    private static final long serialVersionUID = 1L;
    HttpSession session;
    String User = "";
    String Pass = "";

    public ListMail() {
        super();
        // TODO Auto-generated constructor stub
    }

    public void Login(HttpServletRequest req) {
        User = req.getParameter("user");
        Pass = req.getParameter("pass");
        session = req.getSession(true);
        session.setAttribute("userName", User);
        session.setAttribute("passName", Pass);

    }
```

รูปที่ 3.4 Code ส่วนการรับค่าจาก หน้า Login

```
public void Login(HttpServletResponse resp) throws MessagingException,
        IOException {
    resp.setContentType("text/html;charset=UTF-8");
    PrintWriter out = resp.getWriter();

    Folder folder = null;
    Store store = null;

    Properties props = System.getProperties();
    props.setProperty("mail.store.protocol", "imaps");

    Session session = Session.getDefaultInstance(props, null);
    // session.setDebug(true);
    store = session.getStore("imaps");
    store.connect("imap.gmail.com", User, Pass);

    folder = store.getFolder("Inbox");
    folder.open(Folder.READ_WRITE);
    // Message messages[] = folder.getMessages();
    Message messages[] = folder.search(new FlagTerm(new Flags(Flag.SEEN),
            false));// get unread
    int unread = folder.getUnreadMessageCount();
```

รูปที่ 3.5 Code ส่วนการเรียกใช้ IMAP

เมื่อทำการติดต่อกับ IMAP ได้แล้วจึงทำการดึง List Mail ในส่วนของ Mail Unread มาแสดงผลตาม Code ดังรูปที่ 3.6

```
        out.println("<h2>List UnRead Messages</h2>" + " Count : " + unread);
        for (int i = 0; i < messages.length; ++i) {
            Message msg = messages[i];

            String from = "unknown";
            if (msg.getReplyTo().length >= 1) {
                from = msg.getReplyTo()[0].toString();
            } else if (msg.getFrom().length >= 1) {
                from = msg.getFrom()[0].toString();
            }

            out.println("");
            out.println("<br><li>" + " <a href=Content?" + i + " >"
                    + "Subject : " + msg.getSubject() + "</a> - " + from);

        }
        out.println("</div></div></body></head>");

    }
```

รูปที่ 3.6 Code ส่วนแสดง List Mail

List UnRead Messages

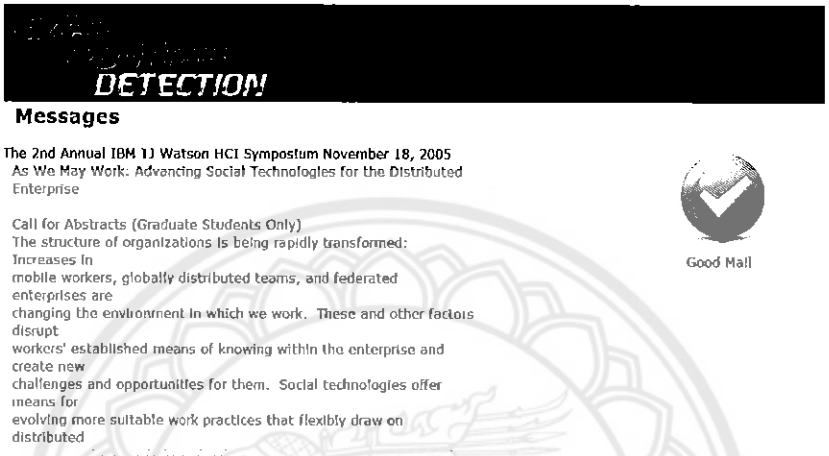
Count : 8

- Subject : , try 3 avast security solutions for FREE - avast Antivirus
- Subject : คณรู้จัก Thirawat Magate, Aye Chalakul และ Mamboo Cpe หรือไม่ - noreply
- Subject : คณรู้จักท่านรองศ. ดุนครับ หรือไม่ - noreply
- Subject : คณรู้จัก Tarn Pronsuang Chucnchom และ Pitchaya Pongchamnan หรือไม่ - noreply
- Subject : คณรู้จัก Direk Hemin, สหพล ลิ่งนีดี และ Porawit Mahachai หรือไม่ - noreply
- Subject : คณรู้จัก นิเรศ แพชรหยอย, Gigi Wathanya และ Naruedon Khunpanya หรือไม่ - noreply

รูปที่ 3.7 แสดงหน้า Inbox

3.2.1.3 ส่วนของการแสดงเนื้อหาของอีเมล

เมื่อทำการเลือกอีเมลที่ต้องการจะตรวจสอบได้แล้ว จะเข้าสู่หน้าแสดงผล โดยจะแสดงเนื้อหา และ ผลการตรวจสอบของอีเมลนั้นดังรูปที่ 3.8



รูปที่ 3.8 แสดงหน้าเนื้อหา

```
out.println("<h2>Messages</h2>");
out.println("<div class=sidebar1>");


int numM = Integer.parseInt(Url);
Message msg = messages[numM];

Object content = msg.getContent();
if (content instanceof String) {
    String body = (String) content;
    //out.println("content : " + body);
} else if (content instanceof Multipart) {
    Multipart multiPart = (Multipart) content;
    int numberOfParts = multiPart.getCount();
    for (int partCount = 0; partCount < numberOfParts; partCount++) {
        BodyPart part = multiPart.getBodyPart(partCount);

        //out.println("content : " + part.getContent().toString());
        st = (String) part.getContent();
    }

}
```

รูปที่ 3.9 Code ส่วนของแสดงหน้าเนื้อหา

ในส่วนของระบบการตรวจสอบนั้น จะให้การตรวจสอบจากเนื้อหาของอีเมล นำมาผ่านการคัดกรองผ่าน โปรแกรม Bayesian filter ที่ประยุกต์จากการตรวจสอบสแปม ดังนี้

SpamFilter.java ส่วนการคัดกรอง

```
public void trainSpam(String file) throws IOException {
            A2ZFileReader fr = new A2ZFileReader(file);

            // Read the content and break up into words
            String content = fr.getContent();
            String[] tokens = content.split(splitregex);
            int spamTotal = 0;// tokenizer.countTokens(); // How many words total

            // For every word token
            for (int i = 0; i < tokens.length; i++) {
                  String word = tokens[i].toLowerCase();
                  Matcher m = wordregex.matcher(word);
                  if (m.matches()) {
                        spamTotal++;
                        // If it exists in the HashMap already
                        // Increment the count
                        if (words.containsKey(word)) {
                              Word w = (Word) words.get(word);
                              w.countBad();
                              // Otherwise it's a new word so add it
                        } else {
                              Word w = new Word(word);
                              w.countBad();
                              words.put(word, w);
                        }
                  }
            }

            // Go through all the words and divide
            // by total words
            Iterator iterator = words.values().iterator();
            while (iterator.hasNext()) {
                  Word word = (Word) iterator.next();
                  word.calcBadProb(spamTotal);
            }
      }
```

```
public void trainGood(String file) throws IOException {
	A2ZFileReader fr = new A2ZFileReader(file);

	// Read the content and break up into words
	String content = fr.getContent();
	String[] tokens = content.split(splitregex);
	int goodTotal = 0;// tokenizer.countTokens(); // How many words total

	// For every word token
	for (int i = 0; i < tokens.length; i++) {
		String word = tokens[i].toLowerCase();
		Matcher m = wordregex.matcher(word);
		if (m.matches()) {
			goodTotal++;
			// If it exists in the HashMap already
			// Increment the count
			if (words.containsKey(word)) {
				Word w = (Word) words.get(word);
				w.countGood();
				// Otherwise it's a new word so add it
			} else {
				Word w = new Word(word);
				w.countGood();
				words.put(word, w);
			}
		}
	}
```

เป็นส่วนของการฝึกโปรแกรมให้อ่านค่าจากไฟล์ที่รวบรวมเอาจดหมายที่ดีและ จดหมายไม่ดีมา วิเคราะห์หาว่าอีเมลที่ได้รับมานั้นใกล้เคียงกับจดหมายดีหรือร้าย จากนั้นเราจะทำไปวิเคราะห์ตามสูตรของ Bayesian ดัง Code ต่อไปนี้

```
public boolean analyze(String stuff) {

	// Create an arraylist of 15 most "interesting" words
	// Words are most interesting based on how different their Spam
	// probability is from 0.5
	ArrayList interesting = new ArrayList();

	// For every word in the String to be analyzed
	String[] tokens = stuff.split(splitregex);

	for (int i = 0; i < tokens.length; i++) {
		String s = tokens[i].toLowerCase();
		Matcher m = wordregex.matcher(s);
		if (m.matches()) {
```

```
                        Word w;

                        // If the String is in our HashMap get the word out
                        if (words.containsKey(s)) {
                              w = (Word) words.get(s);
                                                          // 0.4;
                        } else {
                              w = new Word(s);
                              w.setPSpam(0.4f);
                        }

                        // We will limit ourselves to the 15 most interesting word
                        int limit = 15;
                        // If this list is empty, then add this word in!
                        if (interesting.isEmpty()) {
                              interesting.add(w);
                                                    } else {
                              for (int j = 0; j < interesting.size(); j++) {
                                    // For every word in the list already
                                    Word nw = (Word) interesting.get(j);
                                    // If it's the same word, don't bother
                                    if (w.getWord().equals(nw.getWord())) {
                                          break;

} else if (w.interesting() > nw.interesting()) {
                                          interesting.add(j, w);
                                          break;

} else if (j == interesting.size() - 1) {
                                          interesting.add(w);
                                    }
                              }
                        }

                        // If the list is bigger than the limit, delete entries
                        // at the end (the more "interesting" ones are at the
                        // start of the list
                        while (interesting.size() > limit)
                              interesting.remove(interesting.size() - 1);

                  }
            }

            // Apply Bayes' rule (via Graham)
            float pposproduct = 1.0f;
            float pnegproduct = 1.0f;
            // For every word, multiply Spam probabilities ("Pspam") together
            // (As well as 1 - Pspam)
            for (int i = 0; i < interesting.size(); i++) {
                  Word w = (Word) interesting.get(i);
                  // System.out.println(w.getWord() + " " + w.getPSpam());
                  pposproduct *= w.getPSpam();
                  pnegproduct *= (1.0f - w.getPSpam());
            }
```

```
		// Apply formula
		pspam = pposproduct / (pposproduct + pnegproduct);
		// System.out.println("\nSpam rating: " + pspam);

		// If the computed value is great than 0.9 we have a Spam!!
		if (pspam > 0.9f)
			return true;
		else
			return false;

	}
```

จาก Code โปรแกรมจะคำนวณตามสูตรจะได้ค่าความน่าจะเป็นออกมา จากนั้นโปรแกรมได้ตั้งไว้ว่าเมื่อค่าที่คำนวณออกมานั้นมากกว่า 0.9 จะจัดให้อีเมล์ฉบับนั้นเป็น Spam หรือ Spoof และจะทำการส่งค่าไปให้กับส่วนแสดงผล

ระบบการตรวจสอบโดยคัดกรองเพียงอย่างเดียวนั้นอาจไม่สามารถตรวจสอบได้มากเท่าที่ควร จึงได้มีระบบอีกระบบเพิ่มเข้ามาเพื่อเพิ่มประสิทธิภาพการทำงานให้มากขึ้นคือ การตรวจลิงก์ที่ติดมากับอีเมล โดยจะตรวจสอบกับ List ของลิงก์ดี และลิงก์ที่ไม่ดีโดย List ของลิงก์ดีแล้วไม่ดีนั้นจะรวบรวมมาจากเว็บต่างๆดังนี้

```
public class cLink {

	public ArrayList<String> pullLinks(String text) {
		ArrayList<String> links = new ArrayList<String>();

		String regex = "\\(?\\b(https://|http://|www[.])[-A-Za-z0-
9+&@#/%?=~_()|!:,.;]*[-A-Za-z0-9+&@#/%=~_()|]";
		Pattern p = Pattern.compile(regex);
		Matcher m = p.matcher(text);
		while (m.find()) {
			String urlStr = m.group();
			if (urlStr.startsWith("(") && urlStr.endsWith(")")) {
				urlStr = urlStr.substring(1, urlStr.length() - 1);
			}
			links.add(urlStr);
		}
		return links;
	}

	public int gLinks(String sInput) throws FileNotFoundException {
		int gLink = 0;
		cLink cl = new cLink();
		ArrayList<String> msg = cl.pullLinks(sInput);
		if (msg.size() == 0) {
			gLink = -1;
```

```
		} else {

			File file = new
File("C:\\Users\\GEN\\workspace\\Detection\\WebContent\\WEB-INF\\File\\gLink.txt");
			Scanner scanner = new Scanner(file);
			while (scanner.hasNextLine()) {
				String lineFromFile = scanner.nextLine();
				for (int i = 0; i < msg.size(); i++) {
					String con = msg.get(i).toString();

					if (lineFromFile.contains(con)) {
						// a match!
						gLink++;

					}
				}
			}
			scanner.close();
		}
		return gLink;
	}

	public int bLinks(String sInput) throws FileNotFoundException {
		int bLink = 0;
		cLink cl = new cLink();
		ArrayList<String> msg = cl.pullLinks(sInput);
		if (msg.size() == 0) {
			bLink = -1;

		} else {

			File file = new File(

"C:\\Users\\GEN\\workspace\\Detection\\WebContent\\WEB-INF\\File\\bLink.txt");

			Scanner scanner = new Scanner(file);
			while (scanner.hasNextLine()) {
				String lineFromFile = scanner.nextLine();
				for (int i = 0; i < msg.size(); i++) {
					String con = msg.get(i).toString();

					if (lineFromFile.contains(con)) {
						// a match!
						bLink++;

					}
				}
			}
			scanner.close();
		}
		return bLink;
	}

	}
```

โดยระบบตรวจสอบลิงก์นั้น จะทำการนับค่าของลิงก์ดีแล้วลิงก์ไม่ดีเพื่อมาวิเคราะห์ในลำดับต่อไป

```
SpamFilter filter = new SpamFilter();

filter.trainSpam("C:\\Users\\GEN\\workspace\\Detection\\WebContent\\WEB-INF\\File\\bad.txt");
filter.trainGood("C:\\Users\\GEN\\workspace\\Detection\\WebContent\\WEB-INF\\File\\good.txt");
filter.finalizeTraining();


// Ask the filter to analyze it
boolean spoof = filter.analyze(st);
```

รูปที่ 3.10 Code ส่วนการคัดกรอง

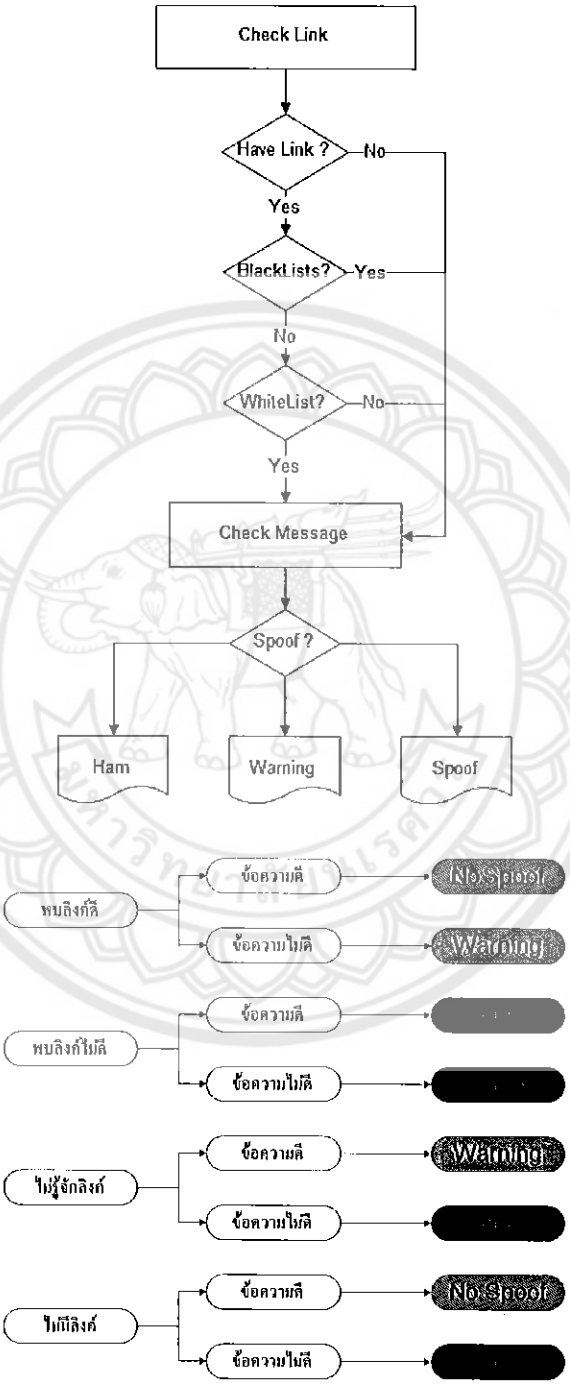
จากรูปที่ 3.10 เป็นการดึงการวิเคราะห์โดยการ Train ค่าจากไฟล์ มาใช้เพื่อคำนวณหาหรือคัดกรองอีเมล์ของแต่ละฉบับ

```
int calc = 0;

cLink cl = new cLink();
if (cl.bLinks(st) > 0) {
        calc = 100;
} else if (cl.gLinks(st) > 0 && cl.bLinks(st) == 0) {
    if(spoof){
        calc = 50;
    }else{
        calc = 0;
    }
} else if(cl.gLinks(st)==-1 && cl.bLinks(st) == -1){
    if(spoof){
        calc = 100;
    }else{
        calc = 0;
    }
}else if(cl.gLinks(st)==0 && cl.bLinks(st) == 0){
    if(spoof){
        calc = 100;
    }else{
        calc = 50;
    }
}
```

รูปที่ 3.11 Code ส่วนการวิเคราะห์ว่าอีเมลที่ได้รับมาเป็น Spoof หรือไม่

จากรูปที่ 3.11 จะเป็นการวิเคราะห์การให้คะแนนของเนื้อหาที่ปลอมแปลงมาหรือไม่ และ ในส่วนของลิงก์ ที่ตรวจสอบมานั้นมีใน Black List หรือไม่ จึงแบ่งออกได้เป็น 8 เงื่อนไขดังนี้



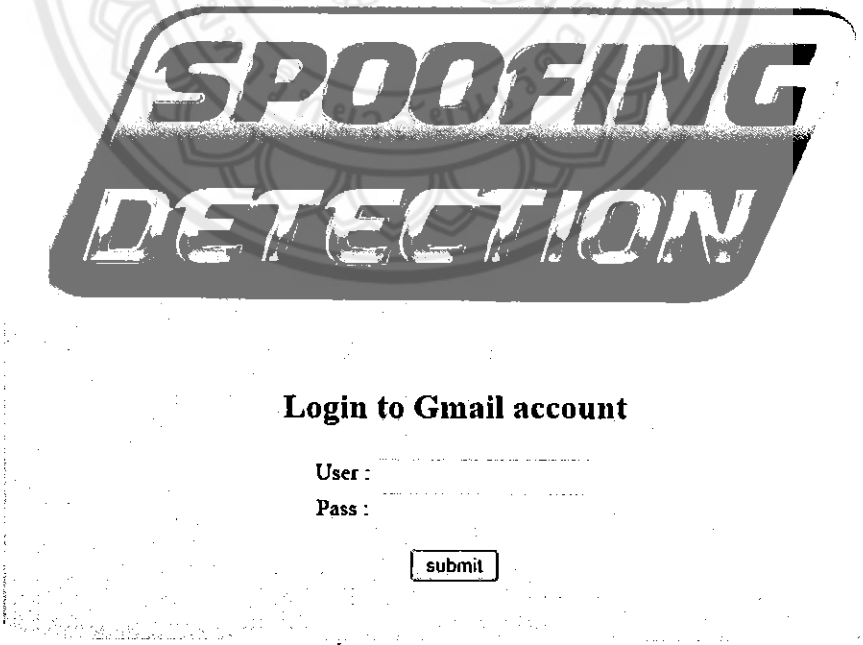
รูปที่ 3.12 เงื่อนไขในการวิเคราะห์อีเมลปลอมแปลง

บทที่ 4
การทดสอบและวิเคราะห์ผล

ทางผู้จัดทำได้ทำการออกโปรแกรม และทำการออกแบบการทดลองโดยนำเอาวิธีการดำเนินงานที่ได้จากบทที่ 3 มาออกแบบเป็นโปรแกรมเพื่อใช้ทดสอบประสิทธิภาพและวิเคราะห์ผลของโปรแกรม

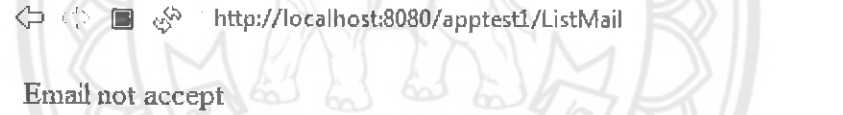
4.1 การทดสอบโปรแกรม

ในการทดสอบโปรแกรมที่นำเสนอ เมื่อโปแกรมเริ่มต้นทำงาน จะต้องทำการกรอก username และ password ให้ถูกต้อง และจะสามารถล็อกอินเข้าสู้โปรแกรมได้ แต่เมื่อทำการกรอก username และ password ผิด จะไม่สามารถทำการล็อกอินก็เข้าสู้โปรแกรมได้

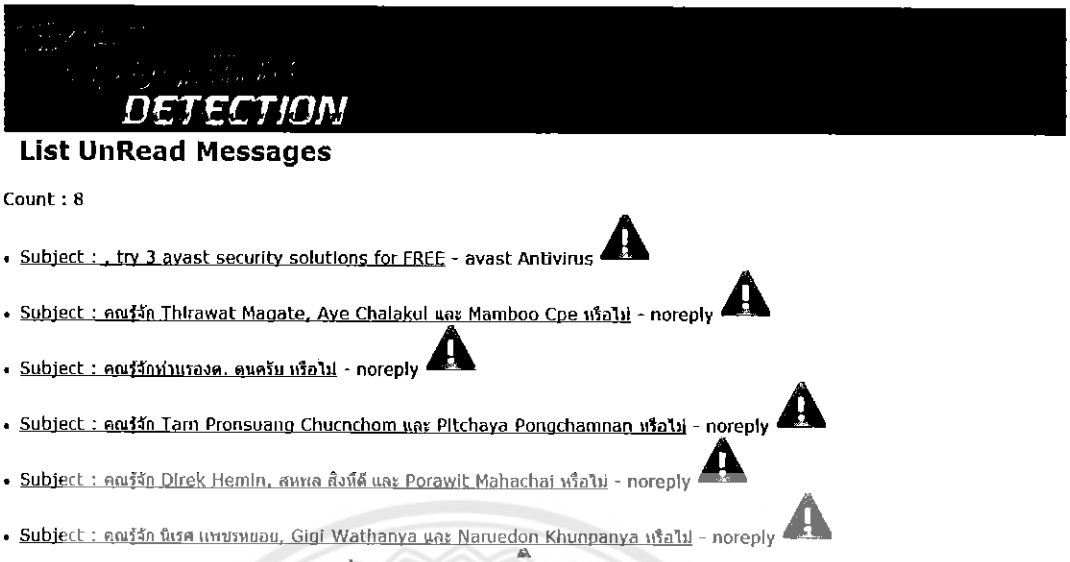


รูปที่ 4.1 โปรแกรมเริ่มต้นทำงาน

รูปที่ 4.2 ทำการกรอก username และ password

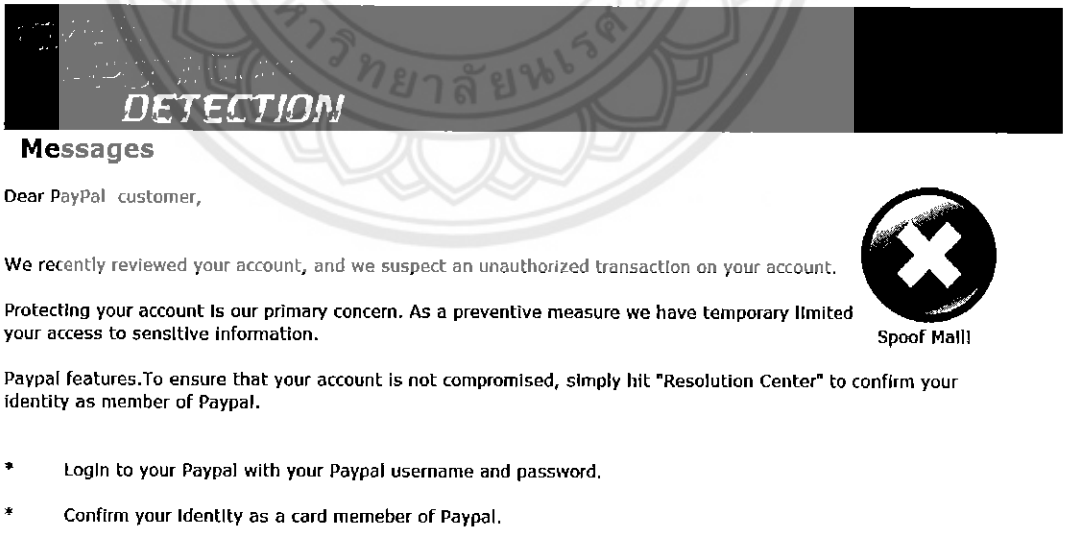


รูปที่ 4.3 ทำการกรอก username และ password ผิด



รูปที่ 4.4 ล็อกอินเข้าสู่โปรแกรมหน้า Inbox

เมื่อทำการล็อกอินเข้าสู่โปรแกรมเรียบร้อยแล้ว โปรแกรมจะแสดงหน้าต่าง Inbox มีการแสดงข้อความอีเมล และจำนวนอีเมลที่ยังไม่ได้ทำการเปิดอ่านกี่ฉบับ เมื่อทำการเปิดอ่านข้อความอีเมล โปรแกรมจะทำการคำนวณ และทำการแสดงว่าอีเมลที่ได้ทำการเปิดอ่านนั้นเป็นอีเมลปลอมแปลง อีเมลที่น่าสงสัย หรืออีเมลปลอดภัย โดยจะทำการแสดงเป็นรูปภาพสัญลักษณ์ และข้อความเตือนขึ้นมา ส่วนอีเมลที่ทำการได้เปิดอ่านแล้ว จะไม่แสดงให้เห็นอีเมลฉบับใน Inbox นั้นอีก



รูปที่ 4.5 แสดงถึงโปรแกรมได้แจ้งเตือนว่าอีเมลฉบับนี้เป็น “อีเมลปลอมแปลง”

Messages

STAR MACHINE CO., LTD.
119/9 Sukhumvit Road, Prakanong
Wattana, Bangkok 10110
Tel. 0 2381 4647-9 Fax. 0 2381 4650
www.orchidresident.com

Waming Mail

10 December 2005

Thai City Electric Co., Ltd.
1643 New Prechburi Road
Makkasan, Rajathewee
Bangkok 10400

Dear Sir,

We are writing in connection with the advertisement for your air conditioners and air purifiers in the Bangkok Post of 9 December 2005. We are the leading manfacturer of machines for the flexible packing industry. We are planning to install air conditioners and air purifiers for our new plant situated in Rayong province.

รูปที่ 4.6 แสดงถึงโปรแกรมได้แจ้งเตือนว่าอีเมลฉบับนี้เป็น "อีเมลน่าสงสัย"

Messages

The 2nd Annual IBM TJ Watson HCI Symposium November 18, 2005
As We May Work: Advancing Social Technologies for the Distributed Enterprise

Call for Abstracts (Graduate Students Only)
The structure of organizations is being rapidly transformed: Increases in
mobile workers, globally distributed teams, and federated enterprises are
changing the environment in which we work. These and other factors disrupt
workers' established means of knowing within the enterprise and create new
challenges and opportunities for them. Social technologies offer means for
evolving more suitable work practices that flexibly draw on distributed

Good Mail

รูปที่ 4.7 แสดงถึงโปรแกรมได้แจ้งเตือนว่าอีเมลฉบับนี้เป็น "อีเมลปลอดภัย"

4.2 วิเคราะห์ผลการทดลอง

ในการวิเคราะห์ประสิทธิภาพของระบบตรวจจับอีเมลปลอมแปลงโดยทั่วไปแล้ว มีปัจจัยในการวิเคราะห์อยู่ 4 ประการ [10] ดังนี้

1. True positive (TP) คือ จำนวนของอีเมลปลอมแปลงที่ถูกวิเคราะห์ว่าเป็นอีเมลปลอมแปลง
2. True negative (TN) คือ จำนวนของอีเมลปกติที่ถูกวิเคราะห์ว่าเป็นอีเมลปกติ
3. False positive (FP) คือ จำนวนของอีเมลปกติที่ถูกวิเคราะห์ว่าเป็นอีเมลปลอมแปลง
4. False negative (FN) คือ จำนวนของอีเมลปลอมแปลงที่ถูกวิเคราะห์ว่าเป็นอีเมลปกติ

จากตัวอย่างอีเมลจำนวน 120 ฉบับ แบ่งออกเป็นอีเมลปลอมแปลงจำนวน 60 ฉบับ และอีเมลปรกติ 60 ฉบับ เมื่อนำมาทำการทดสอบจะได้ผลดัง ตารางที่ 4.2

ตารางที่ 4.1 ผลการวิเคราะห์ของโปรแกรมที่ได้นำเสนอ

True Positive	False Negative	False Positive	True Negative	Average
50	10	19	41	120

นำข้อมูลจากตารางที่ 4.1 มาคำนวณเพื่อหาค่าดังนี้

$$\text{True Positive Rate} = \frac{\text{จำนวนของอีเมลปลอมแปลงที่ถูกวิเคราะห์ว่าเป็นอีเมลปลอมแปลง}}{\text{จำนวนของอีเมลปลอมแปลงทั้งหมด}}$$

$$\text{False Positive Rate} = \frac{\text{จำนวนของอีเมลปรกติที่ถูกวิเคราะห์ว่าเป็นอีเมลปลอมแปลง}}{\text{จำนวนของอีเมลปรกติทั้งหมด}}$$

$$\text{False Negative Rate} = \frac{\text{จำนวนของอีเมลปลอมแปลงที่ถูกวิเคราะห์ว่าเป็นอีเมลปรกติ}}{\text{จำนวนของอีเมลปลอมแปลงทั้งหมด}}$$

$$\text{True Negative Rate} = \frac{\text{จำนวนของอีเมลปรกติที่ถูกวิเคราะห์ว่าเป็นอีเมลปรกติ}}{\text{จำนวนของอีเมลปรกติทั้งหมด}}$$

$$\text{Detection Accuracy Rate} = \frac{\text{จำนวนของอีเมลปลอมแปลงที่ตรวจจับได้ถูกต้อง}}{\text{จำนวนของอีเมลปลอมแปลงที่ตรวจจับได้ทั้งหมด}}$$

Test set

$$\text{True Positive} = \left(\frac{50}{60}\right) \times 100 = 83.33\%$$

$$\text{True Negative} = \left(\frac{41}{60}\right) \times 100 = 68\%$$

$$\text{False Positive} = \left(\frac{19}{60}\right) \times 100 = 31.66\%$$

$$\text{False Negative} = \left(\frac{10}{60}\right) \times 100 = 16.67\%$$

$$\text{Detection Accuracy Rate} = \left(\frac{50}{50+19}\right) \times 100 = 72.46\%$$

จากผลลัพธ์ที่ได้จากการทดสอบ และวิเคราะห์ผลการทดลองดังแสดงในตารางที่ 4.2 แสดงให้เห็นว่า โปรแกรมที่นำเสนอมีประสิทธิภาพในการตรวจจับอีเมลปลอมแปลง ผลจากการวิเคราะห์พบว่า โปรแกรมที่นำเสนอมีอัตราในการตรวจจับอีเมลขยะเฉลี่ยเท่ากับ 83.33% และ อัตราความถูกต้องของการตรวจจับเฉลี่ยเท่ากับ 72.46%

บทที่ 5
สรุปผลการดำเนินงาน

บทนี้จะกล่าวถึงสรุปผลการดำเนินงาน ปัญหาที่พบ ข้อเสนอแนะในการแก้ปัญหา และข้อเสนอแนะสำหรับงานในอนาคตของโครงงาน "การตรวจจับการปลอมแปลงเนื้อหาอีเมล์" เพื่อให้เกิดความเข้าใจในโครงงานและนำไปพัฒนาต่อไป

5.1 สรุปผลการทดลอง

โครงงานชิ้นนี้ได้นำเสนอการตรวจจับการเปลอมแปลงเนื้อหาอีเมล์โดยได้นำแนวคิดของการวิเคราะห์แบบเบย์ ของ Graham [1] ซึ่งนำมาประยุกต์ใช้เพื่อวิเคราะห์เนื้อหาของอีเมล์แต่ละฉบับ จากผลการทดลองในบทที่ผ่านมา ผลลัพธ์ที่ได้ แสดงให้เห็นว่า โปรแกรมที่ได้ทำการนำเสนอมีประสิทธิภาพในการตรวจจับอีเมล์ปลอมแปลง โดยที่สามารถตรวจจับอีเมล์ที่ทำการปลอมแปลงได้ ผลจากการวิเคราะห์พบว่า โปรแกรมที่นำเสนอมีอัตราความถูกต้องของการตรวจจับอีเมล์ปลอมแปลงเฉลี่ยเท่ากับ 83.33%

5.2 อภิปราย

5.2.1 เปรียบเทียบประสิทธิภาพระหว่าง Gmail และโปรแกรมการตรวจจับการปลอมแปลงเนื้อหาอีเมล์ในการตรวจสอบอีเมล์ปลอมแปลง

เมื่อทำการเปรียบเทียบประสิทธิภาพระหว่าง Gmail และโปรแกรมการตรวจจับการปลอมแปลงอีเมล์ โดยใช้ตัวอย่างอีเมล์ปลอมแปลงจำนวน 60 ฉบับ จะได้ผลการทดสอบ ดังนี้

ตารางที่ 5.1 ผลการทดสอบเปรียบเทียบประสิทธิภาพโดยรวม

อีเมล์ปลอมแปลง	Gmail	โปรแกรมการตรวจจับการปลอมแปลงเนื้อหาอีเมล์
พบอีเมล์ปลอมแปลง	25	51
ไม่พบอีเมล์ปลอมแปลง	35	9
รวม	60	60

Gmail ตรวจพบอีเมล์ปลอมแปลง 41%

โปรแกรมตรวจพบอีเมล์ปลอมแปลง 85%

ตารางที่ 5.2 ผลการทดสอบจำนวนอีเมล์ปลอมแปลงที่ Gmail พบ

อีเมล์ปลอมแปลง	จำนวนอีเมล์ปลอมแปลงที่ Gmail พบ	โปรแกรมการตรวจจับการปลอมแปลงเนื้อหาอีเมล์
พบอีเมล์ปลอมแปลง	25	23
ไม่พบอีเมล์ปลอมแปลง	-	2
รวม	25	25

ตารางที่ 5.3 ผลการทดสอบจำนวนอีเมล์ปลอมแปลงที่ Gmail ตรวจไม่พบ

อีเมล์ปลอมแปลง	จำนวนอีเมล์ปลอมแปลงที่ Gmail ตรวจไม่พบ	โปรแกรมการตรวจจับการปลอมแปลงเนื้อหาอีเมล์
พบอีเมล์ปลอมแปลง	-	28
ไม่พบอีเมล์ปลอมแปลง	35	7
รวม	35	35

จากผลการทดสอบเปรียบเทียบประสิทธิภาพระหว่าง Gmail และโปรแกรมการตรวจจับการปลอมแปลงเนื้อหาอีเมล์ในการตรวจสอบอีเมล์ปลอมแปลง จะได้เห็นได้ว่าโปรแกรมการตรวจจับการปลอมแปลงเนื้อหาอีเมล์สามารถตรวจสอบเนื้อหาการอีเมล์ปลอมแปลงได้มากกว่า Gmail

5.2.2 ข้อจำกัดของโปรแกรมการตรวจจับการเปลอมแปลงเนื้อหาอีเมล์

5.2.2.1 ในการตรวจสอบอีเมล์ปลอมแปลง โปรแกรมจำเป็นต้องทำการตรวจสอบคำและเนื้อหาภายในอีเมล์ จึงอาจทำให้ผู้ใช้อีเมล์สูญเสียความเป็นส่วนตัวได้

5.2.2.2 โปรแกรมการตรวจจับการเปลอมแปลงเนื้อหาอีเมล์สามารถตรวจสอบอีเมล์ได้เฉพาะภาษาอังกฤษเท่านั้น ยังไม่สามารถตรวจสอบอีเมล์ที่เป็นภาษาไทย และภาษาอื่นๆได้ จึงอาจทำให้โปรแกรมมีผลการตรวจสอบอีเมล์ปลอมแปลงที่ผิดพลาดได้

5.3 ปัญหาที่พบในการพัฒนา

ในการเทรนโปรแกรมเพื่อให้โปรแกรมสามารถตรวจสอบอีเมล์ว่าเป็นอีเมล์ปลอมแปลง อีเมล์น่าสงสัย หรืออีเมล์ปลอดภัย เนื้อหาในการเทรนนั้นมีมากเกินไป จึงไม่อาจทำให้โปรแกรมสามารถทำการตรวจสอบอีเมล์ได้ไม่เต็มที่

5.4 ข้อเสนอแนะ

5.3.1 ทำให้โปรแกรมสามารถให้ผู้ใช้ทำการเทรนอีเมล์เองได้ ทำให้ตัวโปรแกรมมีความฉลาดมากยิ่งขึ้น

5.3.2 ทำให้โปรแกรมสามารถทำให้โปรแกรมรองรับการใช้งานภาษาไทยได้

5.5 แนวทางการพัฒนาต่อในอนาคต

เนื่องจากวิธีการที่นำเสนอนำเอาเทคนิคที่นำแนวคิดของการวิเคราะห์แบบเบย์ ของ Graham เพื่อใช้ในการตรวจจับอีเมล์ปลอมแปลง จึงส่งผลให้ระบบใช้เวลาและทรัพยากรของเครื่องคอมพิวเตอร์เพิ่มมากขึ้น และยังไม่ได้มีการตรวจสอบตัวตนของผู้ส่งอีเมล์ แนวทางในการพัฒนาต่อในอนาคต ผู้จัดทำจึงมุ่งแก้ไขปัญหาดังกล่าวเพื่อให้ระบบมีความรวดเร็ว ลดการใช้ทรัพยากรของเครื่องคอมพิวเตอร์ และทำการตรวจสอบตัวตนของผู้ส่งอีเมล์ ปรับปรุงเรื่องของฐานข้อมูล Keyword และ Link วิเคราะห์โครงสร้างของอีเมล์ปลอมแปลงแบบอื่นๆว่าเป็นอย่างไร

เอกสารอ้างอิง

[1] "เตือนภัยร้ายจากอีเมล์หลอกลวง" สืบค้นเมื่อ 8 กรกฎาคม 2556, จาก
https://cpc.ku.ac.th/news/51_publicnews/phishingmail.pdf

[2] P. Graham, "A Plan for Spam" สืบค้นเมื่อ 8 กรกฎาคม 2556, จาก
http://www.paulgraham.com/spam.html

[3] "คณิตศาสตร์กับการกำจัดสแปม (Spam)" สืบค้นเมื่อ 11 กรกฎาคม 2556, จาก
http://mathminton.blogspot.com/2011/01/spam.html

[4] Introduction to Java Servlet (in depth) สืบค้นเมื่อ 3 กันยายน 2556, จาก
http://www.jarticles.com/tutorials/servlet/intro_servlet.html

[5] "IMAP คืออะไร" สืบค้นเมื่อ 15 กันยายน 2556, จาก
http://www.com5dow.com/%E0%B9%84%E0%B8%82%E0%B8%9B%E0%B8%B1%E0%B8%8D%E0%B8%AB%E0%B8%B2%E0%B8%A8%E0%B8%B1%E0%B8%9E%E0%B8%97%E0%B9%8C-it/1193-imap-%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3.html

[6] "POP3 กับ IMAP คืออะไร" สืบค้นเมื่อ 15 กันยายน 2556, จาก
http://www.dld.go.th/ict/article/general/gen08.html

[7] "Apache คืออะไร" สืบค้นเมื่อ 3 กันยายน 2556, จาก
http://we-developer.blogspot.com/2009/09/blog-post.html

[8] "Apache Tomcat" สืบค้นเมื่อ 3 กันยายน 2556, จาก
http://th.wikipedia.org/wiki/%E0%B8%AD%E0%B8%B0%E0%B9%81%E0%B8%9E%E0%B8%8A%E0%B8%B5_%E0%B8%97%E0%B8%AD%E0%B8%A1%E0%B9%81%E0%B8%84%E0%B8%95

[9] "Tomcat: Web Container" สืบค้นเมื่อ 3 กันยายน 2556, จาก
http://it-madmonster.blogspot.com/2009/07/web-container.html

[10] C. N. Songkhla and K. Piromsopa, **"Statistical Rules for Thai Spam Detection"**, Proceedings of the Second International Conference on Future Networks 2010, pp.238-242.

[11] "E-mail" สืบค้นเมื่อ 11 มีนาคม 2557, จาก

http://www.com5dow.com/%E0%B9%84%E0%B8%82%E0%B8%9B%E0%B8%B1%E0%B8%8D%E0%B8%AB%E0%B8%B2%E0%B8%A8%E0%B8%B1%E0%B8%9E%E0%B8%97%E0%B9%8C-it/875-e-mail-%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3.html

ประวัติผู้ดำเนินโครงงาน

ชื่อ นายธณวรพงษ์ กี่ดำรงกูล

ภูมิลำเนา 9/123 ถ.มิตรภาพ ต.ในเมือง อ.เมือง

จ.พิษณุโลก 65000

ประวัติการศึกษา

- จบระดับมัธยมศึกษาจากโรงเรียนจ่านกร้อง
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4

สาขาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

Email: jemajern@gmail.com

ชื่อ นายธีรวัต แก้วโชติ

ภูมิลำเนา 305/47 ถ.สิงหวัฒน์ ต.บ้านคลอง อ.เมือง

จ.พิษณุโลก 65000

ประวัติการศึกษา

- จบระดับมัธยมศึกษาจากโรงเรียนจ่านกร้อง
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4

สาขาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

Email: theerawatkc@gmail.com