ระบบป้องกันการเข้าถึงเว็บไซต์ที่ไม่เหมาะสม

Web Site Blocking System

โดย

อภิศิษฎ์ ทนันชัย

รหัสประจำตัว 46066833

อาจารย์ที่ปรึกษา

ผศ.ดร. จันทร์บูรณ์ สถิตวิริยวงศ์

รายงานนี้เป็นส่วนหนึ่งของวิชาโครงการพัฒนาระบบงาน

หลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ

ภาคเรียนที่ 1 ปีการศึกษา 2548

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ชื่อหัวข้อ	ระบบป้องกันการเข้าถึงเว็บไซต์ที่ไม่เหมาะสม
นักศึกษา	นายอภิศิษฏ์ ทนันชัย
อาจารย์ที่ปรึกษา	ผศ.ดร. จันทร์บูรณ์ สถิตวิริยวงศ์
ระดับการศึกษา	วิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ
แขนงวิชา	วิทยาการสารสนเทศ
ปีการศึกษา	2548

บทคัดย่อ

การป้องกันไม่ให้ผู้ใช้งานเว็บบราวเซอร์ได้รับเนื้อหาที่ไม่เหมาะสม โดยทั่วไปแล้วมักจะใช้โปรแกรมที่ทำงานโดยการพิจารณาจากชื่อ URL หรือ IP address ว่าอยู่ในบัญชีดำ (black list) หรือไม่ หากอยู่ก็จะป้องกันไม่ให้ผู้ใช้เข้าไปใช้งานเว็บไซต์นั้นๆ ซึ่งจะสามารถป้องกันได้เฉพาะเว็บไซต์ที่มีผู้ตรวจสอบและนำมากำหนดไว้ในบัญชีดำแล้วเท่านั้น แต่เนื่องจากเว็บไซต์ที่มีเนื้อหาไม่เหมาะสมเกิดขึ้นใหม่ตลอดเวลา การที่จะปรับปรุงรายชื่อของเว็บไซต์เหล่านั้นให้ทันสมัยเสมอจึงเป็นเรื่องที่ทำได้ยาก วิธีการที่ซับซ้อนขึ้นและมีประสิทธิภาพมากกว่า คือการป้องกันโดยใช้โปรแกรมตรวจสอบเนื้อหาทั้งหมดในหน้าเว็บ (content filtering) แล้วจึงค่อยตัดสินใจว่าจะอนุญาตให้มีการใช้งานหรือไม่ แต่โปรแกรมที่มีอยู่ในปัจจุบันมักจะไม่มีการตรวจสอบข้อความที่เป็นภาษาไทย ทำให้เว็บไซต์ที่ไม่เหมาะสมที่มีเนื้อหาข้อความเป็นภาษาไทยหลุดรอดจากการตรวจจับไปได้ โครงการพัฒนาระบบงานนี้จึงพัฒนาการตรวจจับข้อความที่ไม่เหมาะสมที่เป็นภาษาไทยในหน้าเว็บ โดยมีหลักการทำงานคือตัดข้อความที่ไม่ใช่ภาษาไทยออกไปทั้งหมด จากนั้นจึงนำรายการวลีไม่เหมาะสมมาค้นหาในข้อความที่เหลือ ซึ่งวลีไม่เหมาะสมแต่ละวลีจะมีการให้น้ำหนักไว้ด้วย หากหน้าเว็บใดมีน้ำหนักของวลีที่ไม่เหมาะสมมากเกินกว่าที่กำหนดไว้ โปรแกรมก็จะไม่อนุญาตให้มีการใช้งานหน้าเว็บนั้น และแสดงข้อความเตือนกลับมายังเว็บบราวเซอร์ของผู้ใช้

Title	Web Site Blocking System
Student	Mr. Apisit Tananchai
Advisor	Asst. Prof. Dr. Chanboon Sathitwiriyawong
Level of Study	Master of Science in Information Technology
Major	Information Science
Academic Year	2005

ABSTRACT

To protect the web browser users from inappropriate content we usually use the program to filter the URL or IP address of the web server that users try to access. If those URL or IP address matched the one in a black list, the filter program will restrict the access to that web page. But filtering by web URL or IP address is very difficult as sites change and new ones come up all the time. It is impossible to have comprehensive filtering using just URL or IP address. A more complex and effective method is using a program to check every word in the web page containing inappropriate contents and disallow it if it's not suitable. Most programs in the market today do not check for Thai words, so the web pages that have inappropriate Thai contents can escape from the check. This project will develop a system to detect inappropriate Thai content by first remove all characters that is not Thai. Then match the remaining content with the list of weighted inappropriate phrases. If a certain web page has a weight over specified threshold, the program will disallow the access to that page and present a warning message back to the user's web browser.

สารบัญ

หน้า

บทคัดย่อภาษาไทย I

บทคัดย่อภาษาอังกฤษ II

สารบัญ III

สารบัญตาราง VI

สารบัญรูป VII

บทที่

1. บทนำ 1

1.1 ความเป็นมาและความสำคัญของโครงการพัฒนาระบบ 1

1.2 วัตถุประสงค์ของการพัฒนาระบบงาน 2

1.3 ขอบเขตของการพัฒนาโปรแกรม 2

1.4 ประโยชน์ที่คาดว่าจะได้รับ 3

1.5 ขั้นตอนในการพัฒนาระบบงาน 3

1.6 เนื้อหาของแต่ละบท 3

2. ระบบป้องกันการเข้าถึงเว็บไซต์ที่ไม่เหมาะสม 5

2.1 ลักษณะของเว็บไซต์ที่มีเนื้อหาไม่เหมาะสม 5

2.2 รูปแบบของระบบป้องกันการเข้าถึงเว็บไซต์ 6

2.2.1 ระบบป้องกันโดยการตรวจสอบจากบัญชีคำเว็บไซต์ (URL) หรือ IP address 6

2.2.2 ระบบป้องกันโดยการตรวจสอบจากเนื้อหาของหน้าเว็บ (Content filtering) . 7

2.3 โปรแกรมตรวจสอบเนื้อหาของหน้าเว็บ DansGuardian 7

2.3.1 dansguardian 11

2.3.2 OptionContainer 12

2.3.3 FatController 17

2.3.4 FOptionContainer 21

2.3.5 Socket 29

2.3.6 UDSocket 31

สารบัญ (ต่อ)

หน้า

2.3.7 ListManager 33
2.3.8 ListContainer 34
2.3.9 SysV 39
2.3.10 ConnectionHandler 40
2.3.11 DataBuffer 43
2.3.12 DynamicURLList 44
2.3.13 String 45
2.3.14 HTTPHeader 47
2.3.15 HTMLTemplate 49
2.3.16 Ident 50
2.3.17 ImageContainer 51
2.3.18 FDTunnel 51
2.3.19 LanguageContainer 52
2.3.20 RegExp 52
2.3.21 MD5 53
2.3.22 NaughtyFilter 54

3. การออกแบบโปรแกรมป้องกันการเข้าถึงเว็บไซต์ภาษาไทยที่ไม่เหมาะสม 57
3.1 เป้าหมายในการออกแบบโปรแกรม 57
3.2 การออกแบบการทำงานของโปรแกรม 58
3.2.1 ไฟล์ที่ต้องแก้ไขใน DansGuardian 60
3.2.2 ไฟล์ที่ต้องสร้างเพิ่มเติม 62

4. เครื่องมือต่างๆ ที่ใช้ในการพัฒนาโปรแกรม 64
4.1 เครื่องมือที่ใช้ในการพัฒนาโปรแกรม 64
4.2 ขั้นตอนในการพัฒนาโปรแกรม 65

5. การทดสอบโปรแกรม บทสรุป และข้อเสนอแนะ 66

สารบัญ (ต่อ)

หน้า

5.1 การทดสอบการทำงานของโปรแกรม 66
5.2 ขั้นตอนการทดสอบ 67
5.3 สรุป 68
5.4 ข้อเสนอแนะและแนวทางในการพัฒนาต่อ 69
บรรณานุกรม 70
ประวัติผู้เขียน 71

สารบัญตาราง

หน้า

ตารางที่

3.1 ตัวอย่างรหัสตัวอักษรภาษาไทยและลาติน 59
3.2 สรุปไฟล์ต่างๆ ที่ต้องแก้ไขและสร้างเพิ่มเติม 63

สารบัญรูป

หน้า

รูปที่

2.1 ลักษณะการเชื่อมต่อของระบบ 6
2.2 ตัวอย่าง GET method 7
2.3 รูปแบบของการกำหนดค่าใน configuration file 8
2.4 รูปแบบของการกำหนดค่าใน weighted phrase list files 8
2.5 ตัวอย่างของการกำหนดค่าใน weighted phrase list files 9
2.6 ความสัมพันธ์ระหว่าง processes ต่างๆ ของ DansGuardian 10
2.7 การเรียกใช้ DansGuardian 11
2.8 ความสัมพันธ์ของโมดูลต่างๆ ใน DansGuardian 12
3.1 ตัวอย่างวลีที่มีการแทรกตัวอักษรอื่นเข้าไปด้วย 59
3.2 คำสั่งในการเพิ่มพารามิเตอร์ thaifiltering เข้าไปใน dansguardian.conf 60
3.3 คำสั่งในการเพิ่มการตรวจสอบภาษาไทยเข้าไปในไฟล์ weighedphraselist 60
3.4 คำสั่งในการกำหนดค่าให้กับตัวแปรเพื่อตรวจสอบภาษาไทย 61
3.5 คำสั่งเพื่อเพิ่มตัวแปรสำหรับบอกว่าจะมีการตรวจสอบภาษาไทยหรือไม่ 61
3.6 คำสั่งในการเพิ่มการตรวจสอบภาษาไทยเข้าไปในไฟล์ NaughtyFilter.cpp 62
4.1 หน้าจอของโปรแกรม VMwarae 64
5.1 ลำดับการติดต่อกันของโปรแกรมต่างๆ 66
5.2 หน้าเว็บที่มีข้อความไม่เหมาะสม 67
5.3 หน้าเว็บที่มีข้อความไม่เหมาะสมถูกห้ามเข้า 68

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของโครงการพัฒนาระบบ

การใช้งานเครือข่ายอินเทอร์เน็ตในปัจจุบัน เป็นไปอย่างแพร่หลายและเข้าถึงผู้ใช้ทั่วไปได้อย่างสะดวกง่ายดาย เนื่องจากเป็นสื่อกลางในการเข้าถึงแหล่งข้อมูลขนาดใหญ่มากมาย การใช้งานส่วนมากผู้ใช้มักใช้งานผ่านทางเว็บบราวเซอร์ เพื่อติดต่อเข้าถึงข้อมูลต่างๆ ที่อยู่ในเว็บเซิร์ฟเวอร์ ข้อมูลเหล่านี้จะเป็นเอกสารข้อความ รูปภาพ มัลติมีเดีย ในรูปแบบของเอกสาร HTML ด้วยเหตุที่การนำเสนอข้อมูลผ่านทางเว็บเซิร์ฟเวอร์ทำได้โดยง่าย และข้อมูลเหล่านั้นก็สามารถเข้าถึงได้อย่างสะดวกโดยใครก็ได้ ทำให้ผู้ใช้โดยเฉพาะเยาวชนอาจได้รับข้อมูลที่ไม่เหมาะสมกับวัย หรือแม้แต่ผู้ใหญ่เองก็อาจเกิดความรำคาญหรือไม่สบายใจที่ได้รับข้อมูลเหล่านั้น เช่นถ้อยคำหยาบคาย ลามก อนาจาร หรือรุนแรง จึงมีการคิดค้นหาวิธีที่จะป้องกันไม่ให้เว็บไซต์ที่มีข้อความหรือเนื้อหาไม่เหมาะสมผ่านมาถึงผู้ใช้ได้ วิธีการที่ใช้กันทั่วไปในองค์กรคือ การกำหนดบัญชีดำรายชื่อเว็บไซต์ที่ไม่อนุญาตให้ใช้งานไว้ที่พรอกซี่เซิร์ฟเวอร์ และกำหนดให้ผู้ใช้ทุกคนต้องใช้งานเว็บบราวเซอร์ผ่านทางพรอกซี่เซิร์ฟเวอร์เสมอ พรอกซี่เซิร์ฟเวอร์จึงสามารถบังคับไม่ให้ผู้ใช้เข้าถึงเว็บเซิร์ฟเวอร์ที่มีเนื้อหาไม่เหมาะสมได้

การป้องกันไม่ให้ผู้ใช้เข้าถึงเนื้อหาที่ไม่เหมาะสม โดยการกำหนดบัญชีดำรายชื่อของเว็บเซิร์ฟเวอร์ที่ไม่เหมาะสมไว้ที่พรอกซี่เซิร์ฟเวอร์นั้นมีข้อด้อยประการสำคัญก็คือ ต้องมีการปรับปรุงบัญชีดำรายชื่อเว็บเซิร์ฟเวอร์ให้ทันสมัยอยู่เสมอ จึงจะสามารถป้องกันได้อย่างมีประสิทธิภาพ แต่เว็บไซต์ที่มีเนื้อหาไม่เหมาะสมก็เกิดขึ้นใหม่ตลอดเวลา และเป็นไปไม่ได้ที่จะมีบัญชีดำรายชื่อของเว็บเซิร์ฟเวอร์ที่ครบถ้วน ดังนั้นจึงต้องมีวิธีการเสริมที่มีความฉลาดเพียงพอที่จะตรวจสอบเนื้อหาของหน้าเว็บ และตัดสินใจได้เองว่าจะอนุญาตให้ผู้ใช้เข้าถึงเนื้อหาเหล่านั้นได้หรือไม่ โปรแกรมประเภทนี้จะทำงานโดยอาศัยวิธีการหลายๆ อย่างเช่น

- ตรวจสอบวลีที่อยู่ในหน้าเว็บ กับรายการบัญชีดำของวลีที่ไม่เหมาะสม และให้นำหนักหน้าเว็บนั้นว่ามีวลีที่ไม่เหมาะสมมากเกินไปหรือไม่ หากมากเกินกว่าค่าที่กำหนดไว้ก็จะไม่อนุญาตให้มีการใช้งานหน้าเว็บนั้น

- แบ่งกลุ่มของผู้ใช้ออกเป็นหลายๆ กลุ่มเพื่อกำหนดความเข้มข้นของการตรวจสอบให้ไม่เท่ากัน ได้เช่น ตรวจสอบเล็กน้อยสำหรับผู้ใช้ที่เป็นผู้ใหญ่ ตรวจสอบเพิ่มขึ้นสำหรับผู้ใช้วัยรุ่น และตรวจสอบมากที่สุดสำหรับเด็ก
- ตรวจสอบจาก MIME type ของไฟล์ HTML เพื่อป้องกันการเข้าถึงข้อมูลประเภทที่ไม่ต้องการ เช่นวีดีโอ หรือเสียง
- ตรวจสอบโดยใช้ regular expression เพื่อป้องกันการเข้าถึงเว็บไซต์ที่มีชื่อประกอบด้วยวลีที่ไม่เหมาะสมเช่น sex

แต่โปรแกรมที่ทำงานในลักษณะนี้โดยทั่วไป ไม่สามารถตรวจสอบเนื้อหาที่เป็นภาษาไทยได้ เนื่องจากโปรแกรมเหล่านั้นมักจะมีการแปลงตัวอักษรภาษาอังกฤษ หรือตัวอักษรโรมันให้เป็นตัวอักษรตัวเล็กหรือตัวใหญ่ทั้งหมด เพื่อให้ครอบคลุมและง่ายต่อการเปรียบเทียบกับบัญชีดำของวลีที่ไม่เหมาะสม ทำให้หน้าเว็บที่เป็นภาษาไทยถูกเปลี่ยนแปลงข้อความไปโดยไม่ได้ตั้งใจ และไม่สามารถนำบัญชีดำของวลีที่ไม่เหมาะสมที่เป็นภาษาไทยมาใช้ได้ จึงต้องมีการปรับปรุงโปรแกรมให้รู้จักและตรวจสอบวลีที่เป็นภาษาไทย และในขณะเดียวกันก็ต้องมีการสร้างบัญชีดำของวลีที่ไม่เหมาะสมในภาษาไทยเพิ่มเติมเข้ามาด้วย เพื่อให้สามารถตรวจจับและป้องกันการเข้าถึงเว็บไซต์ภาษาไทยที่มีเนื้อหาไม่เหมาะสมได้อย่างถูกต้อง

1.2 วัตถุประสงค์ของการพัฒนาระบบงาน

- ศึกษาหลักการทำงานของโปรแกรม DansGuardian ซึ่งเป็นโปรแกรมที่ทำหน้าที่ตรวจสอบเนื้อหาของหน้าเว็บที่อยู่ภายใต้ลิขสิทธิ์ของ GPL (GNU General Public License) ทำให้สามารถนำมาเป็นแนวทางในการพัฒนาระบบป้องกันการเข้าถึงเว็บไซต์ภาษาไทยที่ไม่เหมาะสมได้
- ออกแบบพัฒนาและปรับปรุงโปรแกรม DansGuardian ให้สามารถรองรับการตรวจสอบวลีที่เป็นภาษาไทยและจัดทำรายการบัญชีดำของวลีภาษาไทยที่ไม่เหมาะสม รวมทั้งให้โปรแกรมสามารถแสดงคำเตือนต่างๆ เป็นภาษาไทยได้

1.3 ขอบเขตของการพัฒนาโปรแกรม

การพัฒนาโปรแกรม DansGuardian ให้สามารถรองรับการใช้งานกับภาษาไทยได้นี้จะแบ่งการทำงานออกเป็นห้าส่วนหลักๆ คือ

- ส่วนของการตัดข้อความที่ไม่ใช่ภาษาไทยออกไปจากหน้าเว็บ เพื่อให้เราสามารถตรวจสอบภาษาไทยได้รวดเร็วยิ่งขึ้น

- ส่วนของการนำบัญชีดำของวลีภาษาไทยมาค้นหาในข้อความภาษาไทย หากพบวลีนั้นๆ ในบัญชีดำก็จะนำค่าน้ำหนักของวลีนั้นมาเพิ่มเข้าเป็นค่าน้ำหนักของหน้าเว็บ หากหน้าเว็บใดมีค่าเกินกว่าที่ควรจะเป็นก็จะไม่อนุญาตให้มีการเข้าถึง
- ส่วนของการสร้างบัญชีดำสำหรับวลีภาษาไทยต่างๆ ที่ไม่เหมาะสม และกำหนดน้ำหนักให้กับแต่ละวลีว่ามีความไม่เหมาะสมมากน้อยต่างๆ กันไป
- ส่วนของการสร้างคำเตือนต่างๆ ให้เป็นภาษาไทย
- ส่วนของการปรับปรุงโปรแกรมหลักของ DansGuardian ให้มาเรียกใช้งานความสามารถในการตรวจสอบภาษาไทยที่ได้เพิ่มเติมเข้ามา

การพัฒนาโปรแกรม DansGuardian ให้รองรับการตรวจสอบหน้าเว็บที่เป็นภาษาไทยนี้จะกระทำภายใต้ระบบปฏิบัติการ Linux (Fedora Core 2) และพัฒนาโดยใช้ภาษา C++

1.4 ประโยชน์ที่คาดว่าจะได้รับ

เพื่อให้องค์กรต่างๆ มีเครื่องมือที่มีความสามารถสูงและรองรับการปกป้องผู้ใช้งานจากการเข้าถึงข้อมูลที่ไม่เหมาะสมที่เป็นภาษาไทยได้ โดยที่ผู้ดูแลระบบไม่จำเป็นต้องคอยปรับปรุงบัญชีดำรายชื่อของเว็บเซิร์ฟเวอร์ที่ไม่ต้องการให้มีการใช้งานอยู่เสมอ และสำหรับสถานศึกษาต่างๆ ก็อาจนำระบบนี้ไปใช้งานเพื่อเยาวชนจากเนื้อหาข้อมูลที่ไม่เหมาะสมที่อยู่บนอินเทอร์เน็ตได้

1.5 ขั้นตอนในการพัฒนาระบบงาน

- ศึกษาหลักการและเทคนิคต่างๆ ในการเขียนโปรแกรมด้วยภาษา C++
- ศึกษาหลักการ เครื่องมือ และเทคนิคต่างๆ ในการเขียนโปรแกรมบนระบบปฏิบัติการ Linux
- ศึกษาหลักการทำงานของโปรแกรม DansGuardian
- วิเคราะห์และออกแบบโปรแกรมตรวจสอบเนื้อหาของหน้าเว็บที่เป็นภาษาไทย
- ทดสอบการใช้งานและปรับปรุงแก้ไขข้อผิดพลาดที่อาจเกิดขึ้น
- สรุปผลการทดสอบ
- จัดทำเอกสารประกอบโครงการ

1.6 เนื้อหาของแต่ละบท

- บทที่ 2 ระบบป้องกันการเข้าถึงเว็บไซต์ที่ไม่เหมาะสม
- บทที่ 3 การออกแบบโปรแกรมป้องกันการเข้าถึงเว็บไซต์ภาษาไทยที่ไม่เหมาะสม
- บทที่ 4 เครื่องมือต่างๆ ที่ใช้งานการพัฒนาโปรแกรม

- บทที่ 5 การทดสอบการใช้งานโปรแกรมและผลที่ได้รับ รวมทั้งบทสรุปและข้อเสนอแนะในการพัฒนาเพิ่มเติมในอนาคต

บทที่ 2
ระบบป้องกันการเข้าถึงเว็บไซต์ที่ไม่เหมาะสม

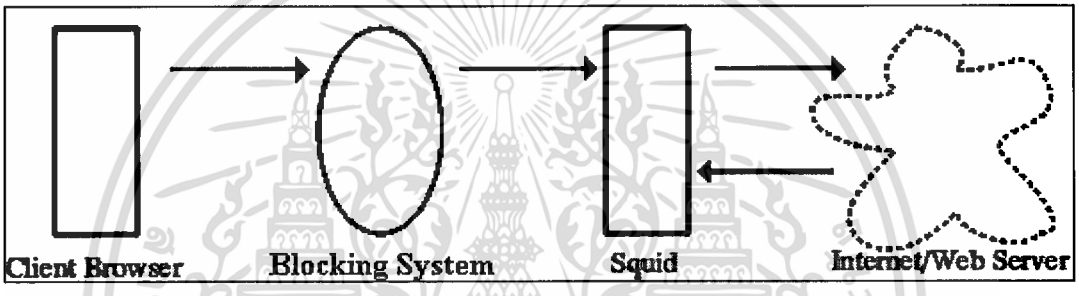
2.1 ลักษณะของเว็บไซต์ที่มีเนื้อหาไม่เหมาะสม

เพื่อให้สามารถออกแบบระบบป้องกันที่มีประสิทธิภาพและครอบคลุมให้ได้มากที่สุด เราจึงจำเป็นต้องรู้ว่าเนื้อหาในเว็บไซต์แบบไหนที่ไม่เหมาะสมหรือเราไม่อยากให้ผู้ใช้ระบบเครือข่ายของเราเข้าไปใช้งาน ซึ่งพอจะแบ่งตามลักษณะของเนื้อหาที่อยู่ภายในหน้าเว็บนั้น ได้ดังตัวอย่างต่อไปนี้

1) Bad words หน้าเว็บที่มีคำหยาบหรือคำสบถสาบานต่างๆ
2) Chat หน้าเว็บที่เกี่ยวกับการพูดคุย สถานศึกษาหรือบริษัทอาจไม่ต้องการให้ผู้ใช้เข้าไปใช้งาน chat room ต่างๆ ก็ได้
3) Drug หน้าเว็บที่เกี่ยวกับยาเสพติด
4) Gambling หน้าเว็บที่เกี่ยวกับการพนันและการเสี่ยงโชค
5) Games หน้าเว็บที่เกี่ยวกับเกมส์ สถานศึกษาหรือบริษัทอาจไม่ต้องการให้ผู้ใช้เข้าไปเล่นเกมส์ในเวลางานก็ได้
6) Search engine หน้าเว็บสำหรับการค้นหา ในที่นี้เราจะป้องกันไม่ให้ผู้ใช้เข้าไปค้นหาคำที่เราไม่ต้องการ เช่นไม่ให้ค้นหาคำว่า hot girls
7) Gore หน้าเว็บที่มีเนื้อหาน่าเกลียดขยะแขยง
8) Illegal drugs หน้าเว็บที่เกี่ยวกับการขายยาที่ไม่ถูกต้อง
9) Intolerance หน้าเว็บที่เหยียดสีผิวหรือเหยียดเชื้อชาติ
10) Nudism หน้าเว็บที่สนับสนุนการเปลื้องผ้า
11) Personals เว็บที่เกี่ยวกับการหาคู่
12) Pornography หน้าเว็บลามกอนาจาร
13) Proxy หน้าเว็บที่ให้บริการพรอกซี่ เพื่อป้องกันไม่ให้ผู้ใช้ของเราหลบไปใช้บริการพรอกซี่เหล่านั้นแทนที่จะผ่านระบบตรวจสอบของเราก่อน
14) Violence หน้าเว็บที่เกี่ยวกับความรุนแรงต่างๆ
15) Warez/hacking หน้าเว็บที่เกี่ยวกับการใช้งานซอฟท์แวร์ผิดกฎหมาย
16) Weapons หน้าเว็บที่เกี่ยวข้องกับอาวุธ

2.2 รูปแบบของระบบป้องกันการเข้าถึงเว็บไซต์

การป้องกันไม่ให้ผู้ใช้งานเว็บบราวเซอร์ในองค์กรเข้าไปใช้เว็บไซต์ที่มีเนื้อหาไม่เหมาะสม องค์กรจะต้องมีระบบที่ควบคุมการเข้าถึงติดตั้งอยู่ ก่อนที่จะทำการเชื่อมต่อเน็ตเวิร์กขององค์กรเข้ากับกับอินเทอร์เน็ต ซึ่งโดยปกติแล้วแต่ละองค์กรจะมีพรอกซี่เซิร์ฟเวอร์เพื่อช่วยเก็บหน้าเว็บที่มีการใช้งานเป็นประจำอยู่แล้ว ดังนั้นระบบป้องกันการเข้าถึงเว็บไซต์ที่ไม่เหมาะสมจึงควรจะอยู่ในตำแหน่งระหว่างเว็บบราวเซอร์กับพรอกซี่เซิร์ฟเวอร์ เพื่อให้มีการบังคับให้การเชื่อมต่อจากผู้ใช้งานเว็บบราวเซอร์ไปยังอินเทอร์เน็ตต้องผ่านระบบนี้ก่อนเสมอ ดังแสดงในรูปที่ 2.1



รูปที่ 2.1 ลักษณะการเชื่อมต่อของระบบ

นอกจากนี้การวางระบบป้องกันนี้ไว้ก่อนหน้าพรอกซี่เซิร์ฟเวอร์ ยังทำให้ระบบสามารถตัดการติดต่อไปยังเว็บเซิร์ฟเวอร์ที่ไม่เหมาะสมได้ทันที หากระบบมีการตรวจสอบจากบัญชีดำรายชื่อเว็บไซต์หรือ IP address โดยไม่จำเป็นต้องมีการติดต่อไปยังเว็บเซิร์ฟเวอร์ก่อน เป็นการลดงานที่จะเกิดที่พรอกซี่เซิร์ฟเวอร์ได้อีกทางหนึ่ง ระบบป้องกันการเข้าถึงเว็บไซต์ที่ไม่เหมาะสมสามารถทำได้สองแบบคือ

2.2.1 ระบบป้องกันโดยการตรวจสอบจากบัญชีดำเว็บไซต์ (URL) หรือ IP address

ระบบป้องกันแบบนี้จะต้องมีการสร้างบัญชีดำรายชื่อเว็บไซต์ที่มีเนื้อหาไม่เหมาะสมขึ้นมาเก็บไว้ที่ระบบป้องกันก่อน การทำงานของระบบจะเริ่มจากเว็บบราวเซอร์ส่งคำร้องขอ (request message) GET method มาให้กับระบบป้องกันดังตัวอย่างในรูปที่ 2.2

```
GET http://www.somehost.com/path/file.html HTTP/1.0
```

รูปที่ 2.2 ตัวอย่าง GET method

ข้อมูลใน GET method จะทำให้ระบบป้องกันทราบได้ว่าชื่อเว็บเซิร์ฟเวอร์ที่ต้องการติดต่อและ URL นั้นๆ อยู่ในบัญชีดำหรือไม่ ซึ่งอยู่ในบัญชีดำ ก็จะส่งข้อความเตือนแจ้งกลับมาให้กับเว็บบราวเซอร์ทราบ และไม่อนุญาตให้เข้าใช้งานเว็บเซิร์ฟเวอร์หรือ URL นั้นๆ

นอกจากนี้ยังอาจตรวจสอบด้วยการใช้ regular expression สำหรับ URL ได้ด้วย เช่นอาจห้ามเข้าใช้งานทุกหน้าเว็บที่มี URL ประกอบด้วยคำว่า sex ทำให้สามารถป้องกันได้ครอบคลุมมากยิ่งขึ้น

2.2.2 ระบบป้องกันโดยการตรวจสอบจากเนื้อหาของหน้าเว็บ (Content filtering)

ระบบป้องกันแบบนี้จะต้องมีการสร้างบัญชีดำรายการของวลีที่ไม่เหมาะสมขึ้นมา และวลีเหล่านี้จะถูกกำหนดน้ำหนักไว้ หน้าเว็บแต่ละหน้าจะถูกนำมาตรวจสอบกับรายการบัญชีดำนี้ ถ้าพบวลีที่อยู่ในบัญชีดำก็จะมีการเพิ่มน้ำหนักของหน้าเว็บนั้นไปเรื่อยๆ และในขณะเดียวกันก็สามารถมีบัญชีขาวสำหรับวลีที่ใช้ลดน้ำหนักของหน้าเว็บลงได้ด้วย เช่นเว็บไซต์ทางสังคมที่ต่อต้านการค้าประเวณีอาจมีคำว่า pornography หรือ porn อยู่ด้วย แต่จริงๆแล้ววลีที่พูดถึงอาจเป็นว่า stop pornography ดังนั้นหน้าเว็บที่มีคำว่า stop pornography อยู่ติดกันก็จะถูกลดน้ำหนักลงได้ทำให้ผ่านระบบตรวจสอบไปได้

หลังจากที่ผ่านการให้น้ำหนักของหน้าเว็บไปแล้ว ก็จะนำน้ำหนักนั้นไปเทียบกับค่าที่กำหนดไว้ ถ้าหน้าเว็บมีน้ำหนักเกินกว่าที่กำหนดก็จะไม่อนุญาตให้มีการเข้าใช้งานหน้าเว็บนั้น และมีข้อความแจ้งเตือนกลับมายังเว็บบราวเซอร์ของผู้ใช้ด้วย

2.3 โปรแกรมตรวจสอบเนื้อหาของหน้าเว็บ DansGuardian

DansGuardian เป็นโปรแกรมที่ทำหน้าที่ตรวจสอบเนื้อหาของหน้าเว็บ ที่อยู่ภายใต้ลิขสิทธิ์ของ GPL (GNU General Public License) สามารถติดตั้งได้กับระบบปฏิบัติการที่เป็น unix ได้เกือบทุกแบบ รวมทั้ง Linux ด้วย โปรแกรม DansGuardian สามารถตรวจสอบเนื้อหาของหน้าเว็บได้แบบ real time และมีการบันทึกเก็บรายละเอียดไว้ใน log file ทำให้นำมาตรวจสอบย้อนหลังได้ว่า

อาจมีผู้ใช้เข้าไปยังหน้าเว็บใดที่ควรจะถูกบล๊อก เพื่อที่จะได้แก้ไขกฎ และรายการบัญชีดำของวลีให้ถูกต้องเหมาะสมยิ่งขึ้น

โปรแกรม DansGuardian ทำงานโดยจะวางตัวอยู่ระหว่างเว็บบราวเซอร์ของผู้ใช้ และพรอกซี่เซิร์ฟเวอร์ ตามที่แสดงในรูปที่ 2.1 โดยที่ผู้ใช้จะต้องกำหนดค่าในเว็บบราวเซอร์ให้ใช้ DansGuardian เป็นพรอกซี่แทน นอกจากนี้การติดตั้งพรอกซี่เซิร์ฟเวอร์ก็ควรจะกำหนดให้เฉพาะ DansGuardian เท่านั้นที่สามารถติดต่อกับพรอกซี่เซิร์ฟเวอร์ได้ เพื่อป้องกันไม่ให้ผู้ใช้งานหลีกเลี่ยงการตรวจจับ โดยการไปติดต่อกับพรอกซี่เซิร์ฟเวอร์เองโดยตรง

โปรแกรม DansGuardian ที่ติดตั้งลงไปในเครื่องเซิร์ฟเวอร์แล้วจะประกอบด้วยส่วนต่างๆ ที่สำคัญสามส่วนคือ

1) ตัวโปรแกรม dansguardian
2) Configuration files ซึ่งมีสองประเภทคือ
   a) dansguardian.conf เป็น configuration file หลักที่ dansguardian จะอ่านขึ้นมากำหนดค่าให้กับตัวแปรต่างๆ
   b) dansguardianfX.conf เป็น configuration file แยกย่อยสำหรับผู้ใช้แต่ละกลุ่ม โดยที่ X จะเป็นตัวเลขที่กำหนดว่าเป็น configuration file ของผู้ใช้กลุ่มใด

   configuration file ทั้งสองประเภทจะมีรูปแบบเหมือนกันดังแสดงในรูปที่ 2.3

```
variable_name = value
```

รูปที่ 2.3 รูปแบบของการกำหนดค่าใน configuration file

3) weighted phrase list files เป็นกลุ่มของไฟล์ที่ใช้สำหรับกำหนดน้ำหนักให้กับวลีต่างๆ ที่พบในหน้าเว็บ

```
<phrase><weight>
<phrase1>,<phrase2><weight>
```

รูปที่ 2.4 รูปแบบของการกำหนดค่าใน weighted phrase list files

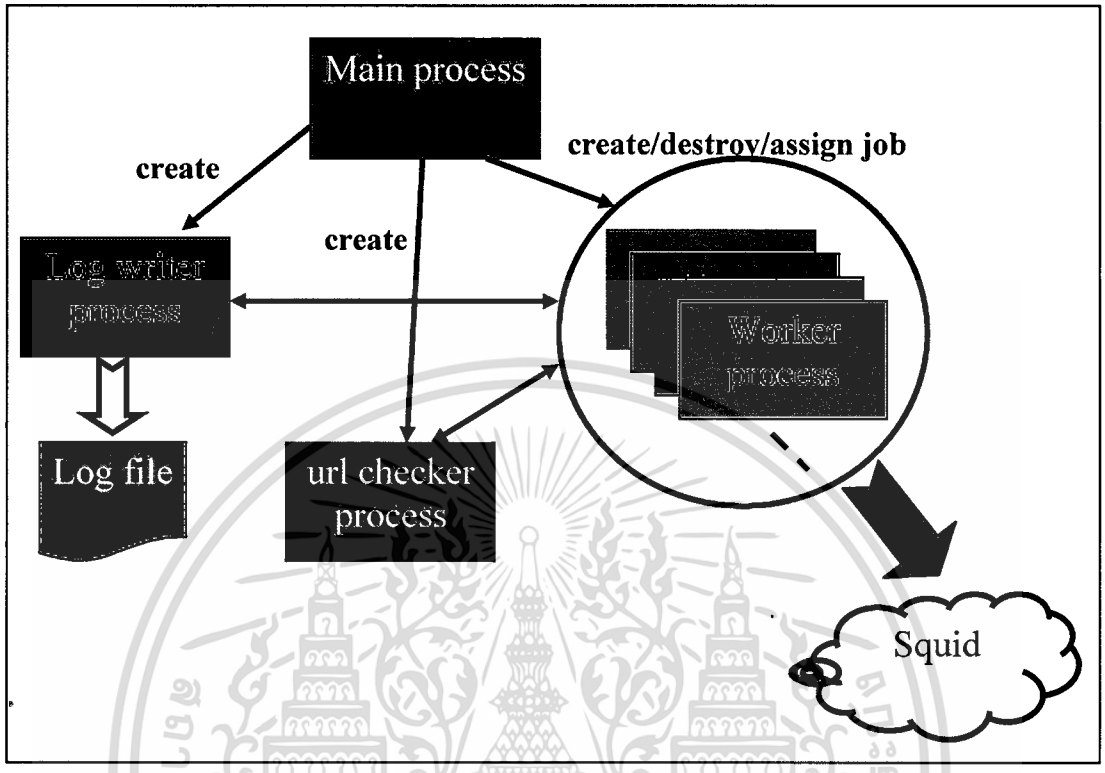
การกำหนดค่าใน weighted phrase list files มีสองแบบดังแสดงในรูปที่ 2.4 โดยที่รูปแบบแรกจะเป็นการเพิ่มค่าน้ำหนัก weight ให้กับหน้าเว็บเมื่อพบวลี phrase ในหน้าเว็บ รูปแบบที่สองจะเป็นการกำหนดค่าน้ำหนัก weight ให้กับหน้าเว็บเมื่อพบทั้งวลี phrase1 และ phrase2 ในหน้าเว็บเดียวกันเท่านั้น โดยต้องระวังว่าทุกตัวอักษรที่อยู่ระหว่าง < และ > จะถูกนับเป็น phrase ทั้งหมดรวมทั้งช่องว่างด้วย และ weight จะสามารถเป็นได้ทั้งค่าบวกและลบ ดังแสดงในรูปที่ 2.5

```
<sex><10> กำหนดค่าให้หน้าเว็บที่มีวลี sex มีน้ำหนัก 10
< sex ><10> กำหนดค่าให้หน้าเว็บที่มีวลี sex (มีช่องว่างหน้าหลังด้วย) มีน้ำหนัก 10
<breast>,<medical><-30> กำหนดค่าให้หน้าเว็บที่มีวลี brease และ medical มีน้ำหนัก -30
<education><-25> กำหนดค่าให้วลี education มีน้ำหนัก -25
```

รูปที่ 2.5 ตัวอย่างของการกำหนดค่าใน weighted phrase list files

เมื่อถูกเรียกขึ้นมาทำงาน DansGuardian จะสร้าง child processes ขึ้นมาสามชนิดด้วยกันดังแสดงในรูปที่ 2.6 ซึ่งมีรายละเอียดคือ

1) log writer process ใช้สำหรับเขียน log file ซึ่งจะรับข้อมูลที่จะเขียนมาจาก worker processes อีกทีหนึ่ง สาเหตุที่ต้องแยกออกมาเป็น process เฉพาะก็เพื่อป้องกันไม่ให้ worker processes แย่งกันเขียน log file
2) url checker process เพื่อใช้สำหรับตรวจสอบว่า url ที่ได้รับมาจากผู้ใช้นั้นเคยถูกตรวจสอบและอนุญาตให้ใช้งานไปแล้วหรือไม่ ทำให้ไม่ต้องมีการตรวจสอบหน้าเว็บซ้ำหลายครั้งถ้ามีผู้ใช้หลายคนต้องการเข้าถึงหน้าเว็บเดียวกัน
3) worker processes เป็นกลุ่มของ processes ที่ทำหน้าที่ในการติดต่อกับผู้ใช้และตรวจสอบหน้าเว็บว่าสมควรจะได้รับอนุญาตให้ผ่านหรือไม่ โดยจะมีการถามไปยัง url checker process ก่อนเพื่อจะได้ไม่ตรวจสอบซ้ำซ้อน และถ้าหน้าเว็บใดไม่ได้รับอนุญาตให้ผ่าน ก็จะส่งรายละเอียดไปให้ log writer process เพื่อเขียนระบุสาเหตุที่ไม่ให้ผ่านลงไปใน log file



รูปที่ 2.6 ความสัมพันธ์ระหว่าง processes ต่างๆ ของ DansGuardian

หลักการทำงานโดยคร่าวๆ ของ DansGuardian ก็คือ main process จะรับการร้องขอหน้าเว็บจากเว็บบราวเซอร์ของผู้ใช้และส่งต่อการร้องขอนี้ไปให้กับ worker process ที่ยังว่างอยู่ จากนั้น worker process ก็จะตรวจสอบว่าผู้ใช้คนนี้ ได้รับอนุญาตให้ใช้งานเว็บหรือไม่ หากได้รับอนุญาตก็จะตรวจสอบกับกฎที่มีอยู่ว่าการร้องขอลักษณะนี้ควรจะได้รับอนุญาตหรือไม่ ถ้าได้รับอนุญาตก็จะส่งการร้องขอนี้ต่อไปยังพรอกซี่เซิร์ฟเวอร์ และรอรับข้อมูลกลับมา จากนั้นจึงตรวจสอบเนื้อหาของข้อมูลข้อมูลนั้นว่ามีวลีหรือข้อความที่ไม่เหมาะสมอยู่หรือไม่ ถ้ามีก็จะแจ้งเตือนให้ผู้ใช้ทราบ แต่ถ้าไม่มีก็จะส่งข้อมูลกลับไปให้ผู้ใช้ นอกจากนี้ยังสามารถกำหนดผู้ใช้ที่มีสิทธิพิเศษ ไม่จำเป็นต้องได้รับการตรวจสอบได้ด้วยซึ่งจะเป็นประโยชน์สำหรับผู้ดูแลระบบ ให้มีความสะดวกคล่องตัวในการทำงานมากยิ่งขึ้น

โปรแกรม DansGuardian พัฒนาด้วยภาษา C++ ประกอบด้วยโมดูลย่อยทั้งหมด 22 โมดูล และทำงานโดยอาศัยค่าเริ่มต้นจาก configuration files ซึ่งมีรายละเอียดของแต่ละโมดูลดังต่อไปนี้

2.3.1 dansguardian

เป็นส่วนเริ่มต้นการทำงานของโปรแกรม หรือฟังก์ชั่น main() ซึ่งจะมีขั้นตอนการทำงานดังนี้

1) รับพารามิเตอร์จาก command line และทำงานตามที่พารามิเตอร์นั้นๆ กำหนด โดยมีการเรียกใช้ดังรูปที่ 2.7

```
dansguardian [{-c ConfigFileName|-v|-P|-h|-N|-q|-s|-r|-g}]
```

รูปที่ 2.7 การเรียกใช้ DansGuardian

พารามิเตอร์แต่ละตัวมีความหมายดังนี้

-c อนุญาตให้สามารถเปลี่ยนไปเรียก configuration file อื่นได้โดยถ้าไม่กำหนดจะเรียกไฟล์ /etc/dansguardian/dansguardian.conf

-v แสดง version ของโปรแกรม

-P แสดงชื่อและ version ของโปรแกรมส่วนขยายอื่นๆ ที่เพิ่มเติมเข้ามา

-h แสดงข้อความช่วยเหลือ

-N ไม่ต้องทำงานเป็น background โดยปกติมักใช้เวลาตรวจสอบข้อผิดพลาด

-q สั่งให้หยุดการทำงานของทุกๆ ก๊อปปี้ ใช้เพื่อ kill ทุกๆ process ของ DansGuardian ที่ทำงานอยู่ในเครื่อง

-s แสดง parent process PID

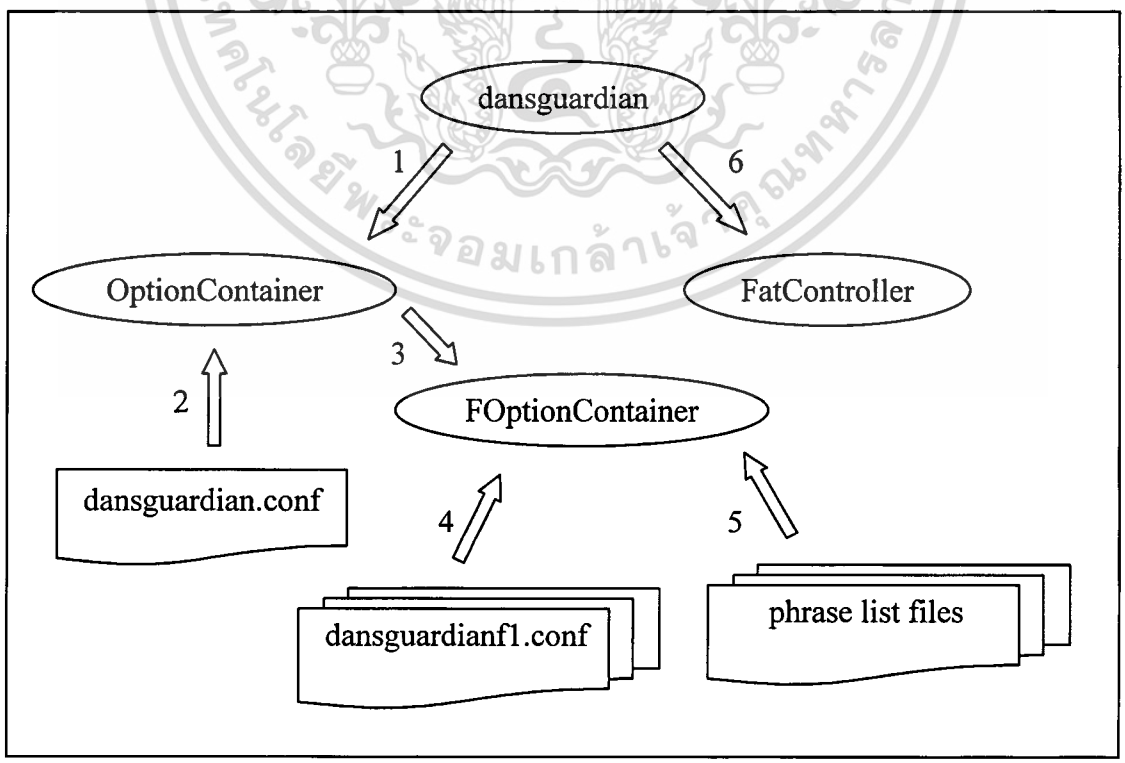
-r ปิด connection ทุกๆ connection และอ่าน configuration file ใหม่ ทำงานโดยการส่ง signal HUP ไปให้ process ที่ทำงานอยู่ แต่ไม่เปลี่ยนแปลงค่า maxchildren

-g เริ่มการทำงานใหม่อย่างนุ่มนวล โดยไม่ปิด connection ที่กำลังค้างอยู่และอ่านค่า filter group file ใหม่เท่านั้น ทำงานโดยการส่ง signal USR1 ไปให้ process ที่ทำงานอยู่

2) อ่านค่า configuration file เข้ามาและกำหนดค่าให้กับตัวแปร o ซึ่งเป็น global variable ใน class OptionContainer สำหรับเก็บค่าตัวแปรต่างๆ จาก configuration file

3) ทำการตรวจสอบว่ามีโปรแกรม DansGuardian ทำงานอยู่ในเครื่องหรือไม่ ถ้ามีก็จะออกจากการทำงานไป เพราะไม่สามารถมีโปรแกรมสองก๊อปปี้ทำงานอยู่พร้อมกันได้

4) ตรวจสอบค่า o.max_children ว่ามากเกินไปหรือไม่ ถ้ามากเกินก็จะออกจากการทำงาน
5) เปิดไฟล์สำหรับเก็บค่า process ID ของโปรแกรมเพื่อใช้ตรวจสอบในภายหลังว่ามีโปรแกรมทำงานอยู่ในเครื่องหรือยัง
6) หาค่า user และ group ของโปรแกรมที่ผู้ใช้กำหนด และเปลี่ยน privilege ไปเป็น user และ group นั้น เพราะโดยปกติแล้วเราจะไม่ให้โปรแกรมทำงานที่ root privilege
7) เข้าสู่ลูปการทำงานหลักซึ่งจะวนรอบการทำงานไปเรื่อยๆ จนกว่าจะเกิดปัญหาหรือถูกสั่งให้หยุดการทำงาน
8) ภายในลูปจะเริ่มจากการตรวจสอบว่า สามารถติดต่อกับพรอกซี่เซิร์ฟเวอร์ได้หรือไม่ ถ้าไม่ได้ก็จะหยุดการทำงาน เนื่องจากเราต้องใช้พรอกซี่เซิร์ฟเวอร์ในการติดต่อไปยังเว็บเซิร์ฟเวอร์ที่ client ต้องการ
9) เรียกฟังก์ชั่น controlIt ซึ่งอยู่ใน class FatController เพื่อให้โปรแกรมทำงานเป็น daemon และ fork child processes ต่างๆ มารับ request จากclient
10) ทำการอ่านค่าจาก configuration file ใหม่ หรือออกจากการทำงาน แล้วแต่ว่าจะถูกสั่งจาก command line ว่าอย่างไร



รูปที่ 2.8 ความสัมพันธ์ของโมดูลต่างๆ ใน DansGuardian

2.3.2 OptionContainer

เป็น class ที่ใช้เก็บตัวแปรหลักต่างๆ ของโปรแกรม และมีฟังก์ชั่นเพื่อจัดการกับตัวแปรเหล่านั้นซึ่งในโปรแกรมหลัก dansguardian.cpp จะกำหนดค่าไว้ในตัวแปร global *o* สำหรับ class OptionContainer มีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ตัวแปรและฟังก์ชั่นแบบ public

OptionContainer() เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า default

~OptionContainer() เป็น destructor ของ class นี้ ใช้ลบค่าต่างๆ จากหน่วยความจำเพื่อป้องกันปัญหา memory leak

*bool read(const char *filename, int type)* เป็นฟังก์ชั่นสำหรับอ่านค่าจาก configuration file ที่กำหนดโดย *filename* และกำหนดค่าให้กับตัวแปรต่างๆ ใน class นี้ ส่วน *type* จะต้องมีค่า 0 หรือ 2 ถ้าเป็น 0 จะกำหนดค่าให้กับตัวแปร 4 ตัวคือ *ipc_filename, urlipc_filename, pid_filename* และ *log_location* เท่านั้น และถ้าเป็น 2 ก็จะกำหนดค่าให้กับตัวแปรอื่นๆ ที่เหลือด้วย

void reset() ใช้สำหรับล้างค่า link lists ต่างๆ ใน class นี้

void deleteFilterGroup() ใช้สำหรับลบ filter group ที่กำหนดไว้ด้วยตัวแปร fg

*bool inipexceptions(const std::string *ip)* ใช้สำหรับตรวจสอบว่า *ip* ที่กำหนดมานั้นอยู่ใน IP exception list หรือไม่

*bool inuserexceptions(const std::string *user)* ใช้สำหรับตรวจสอบว่า *user* ที่กำหนดมานั้นอยู่ใน user exception list หรือไม่

*bool inBannedIPList(const std::string *ip)* ใช้สำหรับตรวจสอบว่า ip ที่กำหนดมานั้นอยู่ใน banned IP list หรือไม่

*bool inBannedUserList(const std::string *user)* ใช้สำหรับตรวจสอบว่า *user* ที่กำหนดมานั้นอยู่ใน banned user list หรือไม่

bool readFilterGroupConf() ใช้สำหรับอ่านค่าของ filter groups จากหลายๆ ไฟล์

bool readfgfile(const char filename)* ถูกเรียกใช้โดย *readFilterGroupConf()* เพื่ออ่านและกำหนดค่าให้กับแต่ละ filter group

int filter_groups ใช้เพื่อบอกว่ามีทั้งหมดกี่ filter group

ตัวแปรต่อไปนี้จะถูกกำหนดค่าจากตัวแปรใน configuration file

int log_exception_hits ถูกกำหนดค่าจากตัวแปร logexceptionhits

int non_standard_delimiter ถูกกำหนดค่าจากตัวแปร nonstandarddelimiter

int log_file_format ถูกกำหนดค่าจากตัวแปร logfileformat

int weighted_phrase_mode ถูกกำหนดค่าจากตัวแปร weightedphrasemode

int show_weighted_found ถูกกำหนดค่าจากตัวแปร showweightedfound

int forwarded_for ถูกกำหนดค่าจากตัวแปร forwardedfor

int createlistcachefiles ถูกกำหนดค่าจากตัวแปร createlistcachefiles

int use_custom_banned_image ถูกกำหนดค่าจากตัวแปร usercustombannedimage

std::string custom_banned_image_file ถูกกำหนดค่าจากตัวแปร custombannedimagefile

ImageContainer banned_image ใช้สำหรับเก็บภาพของ *custom_banned_image_file*

int reverse_lookups ถูกกำหนดค่าจากตัวแปร reverseaddresslookups

int reverse_client_ip_lookups ถูกกำหนดค่าจากตัวแปร reverseclientiplookups

int use_xforwardedfor ถูกกำหนดค่าจากตัวแปร usexforwardedfor

int uim_proxyauth ถูกกำหนดค่าจากตัวแปร usernameidmethodproxyauth

int uim_ntlm ถูกกำหนดค่าจากตัวแปร usenameidmethodntlm

int uim_ident ถูกกำหนดค่าจากตัวแปร usernameidmethodident

int preemptive_banning ถูกกำหนดค่าจากตัวแปร preeemtivebanning

int logconerror ถูกกำหนดค่าจากตัวแปร logconnectionhandlingerrors

int url_cache_number ถูกกำหนดค่าจากตัวแปร urlcachenumber

int url_cache_age ถูกกำหนดค่าจากตัวแปร urlcacheage

int phrase_filter_mode ถูกกำหนดค่าจากตัวแปร phrasefiltermode

int preserve_case ถูกกำหนดค่าจากตัวแปร preservecase

int hex_decode_content ถูกกำหนดค่าจากตัวแปร hexdecodecontent

int force_quick_search ถูกกำหนดค่าจากตัวแปร forcequicksearch

int filter_port ถูกกำหนดค่าจากตัวแปร filterport

int proxy_port ถูกกำหนดค่าจากตัวแปร proxyport

std::string proxy_ip ถูกกำหนดค่าจากตัวแปร proxyip

std::string filter_ip ถูกกำหนดค่าจากตัวแปร filterip

int ll ถูกกำหนดค่าจากตัวแปร loglevel

int reporting_level ถูกกำหนดค่าจากตัวแปร reportinglevel

int max_children ถูกกำหนดค่าจากตัวแปร maxchildren

int min_children ถูกกำหนดค่าจากตัวแปร minchildren

int maxspare_children ถูกกำหนดค่าจากตัวแปร maxsparechildren

int prefork_children ถูกกำหนดค่าจากตัวแปร preforkchildren

int minspare_children ถูกกำหนดค่าจากตัวแปร minsparechildren

int maxage_children ถูกกำหนดค่าจากตัวแปร maxagechildren

std::string daemon_user_name ถูกกำหนดค่าจากตัวแปร daemonuser

std::string daemon_group_name ถูกกำหนดค่าจากตัวแปร daemongroup

int proxy_user ใช้เก็บ user ID ของ dansguardian

int proxy_group ใช้เก็บ group ID ของ dansguardian

int root_user ใช้เก็บ user ID ของ root

std::string filter_groups_list_location ถูกกำหนดค่าจากตัวแปร filtergroupslist

std::string html_template_location ใช้เก็บตำแหน่งของ html template เพื่อแสดงข้อผิดพลาด

std::string banned_ip_list_location ถูกกำหนดค่าจากตัวแปร bannediplist

std::string banned_user_list_location ถูกกำหนดค่าจากตัวแปร banneduserlist

std::string exceptions_user_list_location ถูกกำหนดค่าจากตัวแปร exceptionuserlist

std::string exceptions_ip_list_location ถูกกำหนดค่าจากตัวแปร exceptioniplist

std::string language_list_location ใช้เก็บตำแหน่งของข้อความแสดงข้อผิดพลาด

std::string access_denied_address ถูกกำหนดค่าจากตัวแปร accessdeniedaddress

std::string log_location ถูกกำหนดค่าจากตัวแปร loglocation

std::string ipc_filename ถูกกำหนดค่าจากตัวแปร ipcfilename

std::string urlipc_filename ถูกกำหนดค่าจากตัวแปร urlipcfilename

std::string pid_filename ถูกกำหนดค่าจากตัวแปร pidfilename

int no_daemon ถูกกำหนดค่าจากตัวแปร nodaemon

int no_logger ถูกกำหนดค่าจากตัวแปร nologger

*int soft_restart*ถูกกำหนดค่าจากตัวแปร softrestart

int max_upload_size ถูกกำหนดค่าจากตัวแปร maxuploadsize

int max_content_filter_size ถูกกำหนดค่าจากตัวแปร maxcontentfiltersize

HTMLTemplate html_template ใช้สำหรับเก็บ HTML template ในการใช้งาน

ListContainter filter_groups_list เป็น linked list สำหรับเก็บค่าของ filter group

ListContainter exception_user_list เป็น linked list สำหรับเก็บค่าของ exception user

ListContainter exception_ip_list เป็น linked list สำหรับเก็บค่าของ exception IP

ListContainter banned_ip_list เป็น linked list สำหรับเก็บค่าของ banned IP

ListContainter banned_user_list เป็น linked list สำหรับเก็บค่าของ banned user

LanguageContainer language_list เป็น linked list สำหรับเก็บค่าของข้อความแสดงข้อผิดพลาดเป็นภาษาที่กำหนด

ListManager lm ใช้สำหรับจัดการกับ linked list ทั้งหมดในระบบ

*FOptionContainer** fg* ใช้สำหรับเก็บค่าของ filtergroup

Int numfg ใช้บอกจำนวน filter group ทั้งหมด

ตัวแปรและฟังก์ชั่นแบบ private

std::deque<std::string> conffile เป็น deque สำหรับเก็บตัวแปรต่างๆ ที่กำหนดใน configuration file

String ada ใช้สำหรับเก็บค่า *access_denied_address* โดยเก็บเฉพาะ domain name เท่านั้น

String conffilename ใช้สำหรับเก็บค่าชื่อของ configuration file

bool readbilfile(const char filename)* เป็นฟังก์ชั่นสำหรับอ่านค่า banned IP list จาก *filename*

bool readbuslfile(const char filename)* เป็นฟังก์ชั่นสำหรับอ่านค่า banned user list จาก *filename*

bool readeulfile(const char filename)* เป็นฟังก์ชั่นสำหรับอ่านค่า exception user list จาก *filename*

bool readeilfile(const char filename)* เป็นฟังก์ชั่นสำหรับอ่านค่า exception IP list จาก *filename*

int findoptionI(const char option)* เป็นฟังก์ชั่นสำหรับอ่านค่าตัวแปร *option* ที่เป็น integer จาก deque *conffile*

std::string findoptionS(const char option)* เป็นฟังก์ชั่นสำหรับอ่านค่าตัวแปร *option* ที่เป็น string จาก deque *conffile*

bool realitycheck(String s, int minl, int maxl, char emessage)* เป็นฟังก์ชั่นสำหรับตรวจสอบค่าของ *s* ว่ามีความยาวอยู่ในช่วงที่ควรจะเป็นหรือไม่

*bool readAnotherFilterGroupConf(const char *filename)* เป็นฟังก์ชั่นสำหรับกำหนดค่าให้กับ *fg* และ *numfg*

2.3.3 FatController

เป็น class ที่เปลี่ยนให้โปรแกรมทำงานเป็น daemon และ สร้าง child processes ต่างๆ ขึ้นมาเพื่อรองรับการเรียกใช้งานจาก client ซึ่งในฟังก์ชั่น main() จะกำหนดค่าไว้ในตัวแปร *f* สำหรับ class FatController มีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ตัวแปรและฟังก์ชั่นแบบ public

int controlIt(int pidfilefd) เป็นฟังก์ชั่นสำหรับควบคุมการทำงานให้เป็น daemon (main process ในรูปที่ 2.6) และสร้าง child process (worker process ในรูปที่ 2.6) ขึ้นมาเพื่อรองรับการเรียกใช้งานจาก client มีรายละเอียดการทำงานดังนี้

1) ล้างค่าตัวแปรที่เกี่ยวข้องกับ socket
2) สร้าง server socket สำหรับรองรับการเรียกใช้งานจาก client โดย bind เข้ากับ o.*filter_ip* และ o.*filter_port*
3) ลด privilege ลงเป็นตามที่กำหนดโดย o.*proxy_user*
4) สร้าง ipc socket สำหรับใช้ในการติดต่อกับ child processes และ bind เข้ากับไฟล์ *o.ipc_file_name*
5) สร้าง url list socket สำหรับใช้ในการติดต่อกับ child processes และ bind เข้ากับไฟล์ *o.urlipc_filename*
6) สร้าง peer socket สำหรับใช้ในการติดต่อกับพรอกซี่เซิร์ฟเวอร์
7) เปลี่ยนการทำงานเป็น daemon โดยเรียกใช้งานฟังก์ชั่น *daemonise()*
8) กำหนดให้โปรแกรม ignore signal SIGPIPE และ SIGHUP
9) Fork child process สำหรับเขียนข้อมูลลงใน log file เมื่อได้รับการติดต่อจาก client (log write process ในรูปที่ 2.6) โดยเรียกใช้งานฟังก์ชั่น *logListener()*
10) Fork child process สำหรับจัดการกับ URL list cache (url checker process ในรูปที่ 2.6) โดยเรียกใช้งานฟังก์ชั่น *urlListListener()*
11) กำหนดฟังก์ชั่น signal handler ให้กับ signal SIGTERM, SIGHUP และ SIGUSR1

12) กำหนดค่าเริ่มต้นให้กับตัวแปร *numchildren, busychildren, freechildren, childrenpids childrenstates* และ *pids*

13) Fork child process สำหรับติดต่อกับ client ตามจำนวนที่กำหนดโดย o.min_children โดยเรียกใช้งานฟังก์ชั่น *preFork()*

14) เข้าสู่ลูปการทำงานหลักของ parent process ที่จะควบคุมการทำงานทั้งหมด โดยที่จะหลุดจากลูปหลักนี้เมื่อถูกสั่งให้เริ่มการทำงานใหม่ หรือ มีข้อผิดพลาดเกิดขึ้นติดต่อกันเกิน 30 ครั้ง หรือ ถูกสั่งให้อ่านค่า filter group ใหม่

15) ตรวจสอบว่าจำเป็นต้องอ่านค่า filter group file ใหม่หรือไม่ (ซึ่งจะถูกกำหนดจาก command line option –g)

16) ถ้ามี child process exit ก็ปรับค่าตัวแปร *childrenpid, busychildren, childrenstates, pids* ให้ถูกต้องโดยการเรียกใช้ฟังก์ชั่น *mopUpAfterKids()*

17) ตรวจสอบว่ามีการติดต่อมาจาก client หรือไม่ ถ้ามีก็ส่งให้กับ child process ที่ยังว่างอยู่ ถ้า child process ไม่ว่างก็ให้ตรวจสอบว่าจำนวน child process เกินกว่าที่กำหนดไว้โดย *o.max_children* หรือยัง ถ้ายังก็ fork child process เพิ่มเพื่อรับการติดต่อจาก client

18) ตรวจดูว่ามี child process ทำงานอยู่เกินกว่าที่กำหนดไว้ด้วย o.maxspare_children หรือไม่ ถ้ามีอยู่เป็นเวลาเกินกว่า 120 วินาทีก็ให้ kill child process ที่เกินนั้นไปด้วยฟังก์ชั่น *cullChildren()*

19) ถ้าหลุดออกจากลูปหลัก (จาก 14) แล้วให้ kill child process ทั้งหมดและล้างค่าของตัวแปร *childrenpid, childrenstates, pids*

20) คืนค่า signal handler เดิมให้กับ SIGTERM, SIGHUP, SIGUSR1, SIGPIPE

21) Kill child process สำหรับเขียน log file (จาก 9) และ child process สำหรับจัดการกับ URL list cache (จาก 10)

bool testProxy(std::string proxyip, int proxyport, bool report) ใช้สำหรับตรวจสอบว่าสามารถติดต่อกับพรอกซี่เซิร์ฟเวอร์ตามที่กำหนดโดย *proxyip* และ *proxyport* ได้หรือไม่

ตัวแปรและฟังก์ชั่นแบบ private

int logListener(std::string log_location, int logconerror) เป็นฟังก์ชั่นการทำงานในส่วนของ child process ที่ใช้ในการเขียน log file โดยจะเป็นเพียง process เดียวที่รับคำสั่งจาก processes อื่นๆ ที่ต้องการเขียน log file ผ่านทาง ipc socket (ข้อ 4 ของฟังก์ชั่น controlIt)

bool daemonise(int pidfilefd) เป็นฟังก์ชั่นที่ใช้เปลี่ยนการทำงานของโปรแกรมให้เป็น daemon และเขียนค่า process ID ลงใน *pidfilefd*

int urlListListener(int logconerror) เป็นฟังก์ชั่นการทำงานในส่วนของ child process ที่ใช้ในการจัดการกับ URL list cache โดยจะเป็น process ที่รับคำสั่งจาก processes อื่นๆ ที่ต้องการจัดการกับ URL list cache ผ่านทาง url list socket (ข้อ 5 ของฟังก์ชั่น controlIt) การจัดการกับ URL list cache มี 3 กรณีคือ

1) ได้รับข้อมูลที่ขึ้นต้นด้วย 'F' และตามด้วย space จะเป็นการ flush URL list cache
2) ได้รับข้อมูลที่ขึ้นต้นด้วย 'A' และตามด้วย space จะเป็นการเพิ่ม URL เข้าไปใน URL list cache
3) ถ้าไม่ใช่ทั้งสองกรณีแรกจะนำข้อมูลที่ได้รับไปค้นหาใน URL list cache ดูว่ามีอยู่หรือไม่ ถ้ามีจะตอบ "Y\n" กลับไป แต่ถ้าไม่มีจะตอบ "N\n"

int preFork(int num) เป็นฟังก์ชั่นที่ใช้ fork child process สำหรับจัดการกับการติดต่อจาก client ตามจำนวน *num* ที่กำหนด

void mopUpAfterKids() เป็นฟังก์ชั่นที่จะเรียกฟังก์ชั่น *deleteChild()* เมื่อพบว่ามี child process exit

bool checkForInput(int fd, int timeout) เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่ามีข้อมูลส่งเข้ามาที่ descriptor (หรือ socket) *fd* หรือไม่ โดยจะรอเป็นเวลา *timeout* วินาที

bool checkKidReadyStatus(int tofind) เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่ามี child process ว่างพร้อมที่จะทำงานหรือไม่

int getFreeChild() เป็นฟังก์ชั่นที่ใช้ return ค่า child process ที่ว่างอยู่

void tellChildAccept(int num) เป็นฟังก์ชั่นที่ใช้บอกให้ child process หมายเลข *num* รับการติดต่อโดย parent process จะส่ง "G\n" ไปให้

void cullChildren(int num) เป็นฟังก์ชั่นที่ใช้ส่ง SIGTERM ไปให้ child processes จำนวน *num* processes

void killAllChildren() เป็นฟังก์ชั่นที่ใช้ส่ง SIGTERM ไปให้ child processes ทุก process

*void hupAllChildren()*เป็นฟังก์ชั่นที่ใช้ส่ง SIGHUP ไปให้ child processes ทุก process

void tidyUpForChild() เป็นฟังก์ชั่นที่ใช้คืน signal handler ให้กับ signal SIGTERM, SIGUSR1 และ SIGHUP

int handleConnections(int pipe) เป็นฟังก์ชั่นที่ parent process เรียกใช้ child process ให้จัดการกับการติดต่อจาก client ผ่านทาง *pipe* โดยที่ child process จะมีการเรียกใช้งานฟังก์ชั่น *handleConnection* จาก class ConnectionHandler

void addChild(int pos, int fd, pid_t child_pid) เป็นฟังก์ชั่นที่ใช้กำหนดค่าตัวแปร *childrenpids, childrenstates, numchildren, busychildren, waitingfor* และ *pids* ให้ถูกต้อง

int sendReadyStatus(int pipe) เป็นฟังก์ชั่นที่ child processใช้ส่งค่า ready status กลับไปให้ parent process โดยการส่งค่า "2\n" ไปให้

bool getFDFromParent(int fd) เป็นฟังก์ชั่นที่ child process ใช้สำหรับรับการติดต่อจาก parent process ที่จะส่งค่า "G\n" มาให้เป็นการบอกว่ามี client ติดต่อเข้ามา และ child process จะส่ง "K\n" กลับไปให้ parent process เพื่อบอกว่าพร้อมรับข้อมูลจาก client แล้ว

int getChildSlot() เป็นฟังก์ชั่นที่ใช้สำหรับหาที่ว่างใน *childrenpids*

int getLineFromFD(int fd, char buff, int size, int timeout)* เป็นฟังก์ชั่นที่ใช้สำหรับอ่านค่าจาก *fd* เข้ามาเก็บไว้ใน *buff* ที่มีขนาด *size* และรอเป็นเวลาไม่เกิน *timeout* วินาที

void deleteChild(int child_pid, int stat) เป็นฟังก์ชั่นที่ใช้กำหนดค่า *childrenpids, childrenstates, busychildren* และ *numchildren* ให้ถูกต้อง จะถูกเรียกใช้เมื่อมีการ kill child process

bool writeToFd(int fd, const char buff, int len, int timeout)* เป็นฟังก์ชั่นที่ใช้สำหรับส่งค่า *buff* ขนาด *len* ไปให้ *fd* โดยรอเป็นเวลา *timeout* วินาที

bool readyForOutput(int fd, int timeout) เป็นฟังก์ชั่นที่ใช้เพื่อตรวจสอบว่า *fd* พร้อมที่จะรับข้อมูลหรือยัง โดยจะรอเป็นเวลา *timeout* วินาที

*int selectEINTR(int numfds, fd_set * redfds, fd_set * writefds, fd_set * exceptfds, struct timeval * timeout)* เป็นฟังก์ชั่นที่ใช้เรียกฟังก์ชั่น select โดยจะมีการเรียกซ้ำถ้าได้รับ signal

*bool dropPrivCompletely()*เป็นฟังก์ชั่นที่ใช้ลด privilege ลงมาเป็น *o.proxy_user*

*void flushURLCache()*เป็นฟังก์ชั่นที่ใช้สำหรับล้างค่า URL list cache โดยการส่งค่า "F \n" ไปให้ child process ที่จัดการกับ URL list cache (urlListListener)

int numchildren เป็นตัวแปรที่ใช้บอกว่าตอนนี้มี child process อยู่กี่ process

int busychildren เป็นตัวแปรที่ใช้บอกว่าตอนนี้มี child process ที่กำลังทำงานอยู่กี่ process

int freechildren เป็นตัวแปรที่ใช้บอกว่าตอนนี้มี child process ที่ว่างอยู่กี่ process

int waitingfor เป็นตัวแปรที่ใช้บอกว่าตอนนี้มี child process ที่กำลังรอจะ prefork อยู่กี่ process

int childrenpids* เป็นตัวแปรที่ใช้เก็บ pid ของ child processes, มีค่าเป็น -1 ถ้าไม่ได้ใช้งาน

int childrenstates* เป็นตัวแปรที่ใช้เก็บสถานะของ child process โดยมีค่าเป็น

-2 ถ้า child process กำลังถูก kill โดย SIGHUP หรือ SIGTERM

-1 ถ้าไม่ได้ใช้งาน

0 ถ้า child process ว่างอยู่

1 ถ้า child process กำลังทำงานอยู่

4 ถ้า child process ไม่ว่างและกำลังรอการกำหนดค่าเริ่มต้นอยู่

struct pollfd pids* เป็นตัวแปรที่ใช้สำหรับเก็บ pid ของ child processes

int failurecount เป็นตัวแปรที่ใช้สำหรับเก็บจำนวนข้อผิดพลาดที่เกิดขึ้น

Socket serversock เป็นตัวแปรที่ใช้สำหรับเก็บ socket ที่ใช้สำหรับรับการติดต่อจาก client

UDSocket ipcsock เป็นตัวแปรที่ใช้สำหรับเก็บ socket ที่ใช้สำหรับติดต่อกับ child process ที่ใช้เขียน logfile

UDSocket urllistsock เป็นตัวแปรที่ใช้สำหรับเก็บ socket ที่ใช้สำหรับติดต่อกับ child process ที่ใช้จัดการกับ URL list cache

Socket peersock เป็นตัวแปรที่ใช้สำหรับเก็บ socket ที่ใช้สำหรับติดต่อกับ client

int peersockfd เป็นตัวแปรที่ใช้สำหรับเก็บ descriptor ที่ใช้สำหรับติดต่อกับ client

String peersockip เป็นตัวแปรที่ใช้สำหรับเก็บ IP address ที่ใช้ติดต่อกับ client

int peersockport เป็นตัวแปรที่ใช้สำหรับเก็บ port ที่ใช้ติดต่อกับ client

2.3.4 FOptionContainer

เป็น class สำหรับเก็บและจัดการกับ filter group ซึ่งกำหนดอยู่ใน class OptionContainer โดยตัวแปร *fg* สำหรับ class FOptionContainer มีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ตัวแปรและฟังก์ชั่นแบบ public

~FOptionContainer() เป็น destructor ของ class นี้ ใช้ลบค่าต่างๆ จากหน่วยความจำเพื่อป้องกันปัญหา memory leak

bool read(std::string filename) เป็นฟังก์ชั่นสำหรับอ่านค่า configuration file ของแต่ละ filter group และกำหนดค่าให้กับตัวแปรอื่นๆ ใน class นี้

void reset() เป็นฟังก์ชั่นสำหรับล้างค่าของ linked list ต่างๆ ใน class นี้

bool inexceptions(String url) เป็นฟังก์ชั่นสำหรับตรวจสอบว่า domain *url* อยู่ใน exception site list หรือไม่

bool iswebserver(String url) เป็นฟังก์ชั่นสำหรับตรวจสอบว่า *url* อยู่ในเว็บเซิร์ฟเวอร์ของเราหรือไม่

bool inurlexceptions(String url) เป็นฟังก์ชั่นสำหรับตรวจสอบว่า *url* อยู่ใน exception URL list หรือไม่

char inBannedSiteList(String url)* เป็นฟังก์ชั่นสำหรับตรวจสอบว่า *url* อยู่ใน banned site list หรือไม่

char inBannedURLList(String url)* เป็นฟังก์ชั่นสำหรับตรวจสอบว่า *url* อยู่ใน banned URL list หรือไม่

bool inGreySiteList(String url) เป็นฟังก์ชั่นสำหรับตรวจสอบว่า *url* อยู่ใน grey site list หรือไม่

bool inGreyURLList(String url) เป็นฟังก์ชั่นสำหรับตรวจสอบว่า *url* อยู่ใน grey URL list หรือไม่

int inBannedRegExpURLList(String url) เป็นฟังก์ชั่นสำหรับตรวจสอบว่า *url* อยู่ใน banned regular expression list หรือไม่

char inBannedExtensionList(String url)* เป็นฟังก์ชั่นสำหรับตรวจสอบว่า *url* อยู่ใน banned extension list หรือไม่

bool isIPHostname(String url) เป็นฟังก์ชั่นสำหรับตรวจสอบว่า *url* เป็น IP hostname หรือไม่

int weighted_phrase_mode ถูกกำหนดค่าจากตัวแปร weightedphrasemode ใน configuration file

int naughtyness_limit ถูกกำหนดค่าจากตัวแปร naughtynesslimit ใน filter group configuration file

int createlistcachefiles ถูกกำหนดค่าจากตัวแปร createlistcachefiles ใน configuration file

int enable_PICS ถูกกำหนดค่าจากตัวแปร enablePICS ใน picsfile

int blanketblock เป็นตัวแปรสำหรับเก็บค่าว่าต้องการทำ blanket block หรือไม่

int blanket_ip_block เป็นตัวแปรสำหรับเก็บค่าว่าต้องการทำ blanket IP block หรือไม่

int reverse_lookups ถูกกำหนดค่าจากตัวแปร reverseaddresslookups ใน configuration file

int force_quick_search ถูกกำหนดค่าจากตัวแปร forcequicksearch ใน configuration file

int bypass_mode ถูกกำหนดค่าจากตัวแปร bypass ใน filter group configuration file

int pics_rsac_violence ถูกกำหนดค่าจากตัวแปร RSACviolence ใน picsfile

int pics_rsac_sex ถูกกำหนดค่าจากตัวแปร RSACsex ใน picsfile

int pics_rsac_nudity ถูกกำหนดค่าจากตัวแปร RSACnudity ใน picsfile

int pics_rsac_language ถูกกำหนดค่าจากตัวแปร RSAClanguage ใน picsfile

int pics_icra_chat ถูกกำหนดค่าจากตัวแปร ICRAchat ใน picsfile

int pics_icra_moderatedchat ถูกกำหนดค่าจากตัวแปร ICRAmoderatedchat ใน picsfile

int pics_icra_languagesexual ถูกกำหนดค่าจากตัวแปร ICRAlanguagesexual ใน picsfile

int pics_icra_languageprofanity ถูกกำหนดค่าจากตัวแปร ICRAlanguageprofanity ใน picsfile

int pics_icra_languagemildexpletives ถูกกำหนดค่าจากตัวแปร ICRAlanguagemildexpletives ใน picsfile

int pics_icra_nuditygraphic ถูกกำหนดค่าจากตัวแปร ICRAnuditygraphic ใน picsfile

int pics_icra_nuditymalegraphic ถูกกำหนดค่าจากตัวแปร ICRAnuditymalegraphic ใน picsfile

int pics_icra_nudityfemalegraphic ถูกกำหนดค่าจากตัวแปร ICRAnudityfemalegraphic ใน picsfile

int pics_icra_nuditytopless ถูกกำหนดค่าจากตัวแปร ICRAnuditytopless ใน picsfile

int pics_icra_nuditybottoms ถูกกำหนดค่าจากตัวแปร ICRAnuditybottoms ใน picsfile

int pics_icra_nuditysexualacts ถูกกำหนดค่าจากตัวแปร ICRAnuditysexualacts ใน picsfile

int pics_icra_nudityobscuredsexualacts ถูกกำหนดค่าจากตัวแปร ICRAnudityobscuredsexualacts ใน picsfile

int pics_icra_nuditysexualtouching ถูกกำหนดค่าจากตัวแปร ICRAnuditysexualtouching ใน picsfile

int pics_icra_nuditykissing ถูกกำหนดค่าจากตัวแปร ICRAnuditykissing ใน picsfile

int pics_icra_nudityartistic ถูกกำหนดค่าจากตัวแปร ICRAnudityartistic ใน picsfile

int pics_icra_nudityeducational ถูกกำหนดค่าจากตัวแปร ICRAnudityeducational ใน picsfile

int pics_icra_nuditymedical ถูกกำหนดค่าจากตัวแปร ICRAnuditymedical ใน picsfile

int pics_icra_drugstobacco ถูกกำหนดค่าจากตัวแปร ICRAdrugstobacco ใน picsfile

int pics_icra_drugsalcohol ถูกกำหนดค่าจากตัวแปร ICRAdrugsalcohol ใน picsfile

int pics_icra_drugsuse ถูกกำหนดค่าจากตัวแปร ICRAdrugsuse ใน picsfile

int pics_icra_gambling ถูกกำหนดค่าจากตัวแปร ICRAgambling ใน picsfile

int pics_icra_weaponuse ถูกกำหนดค่าจากตัวแปร ICRAweaponuse ใน picsfile

int pics_icra_intolerance ถูกกำหนดค่าจากตัวแปร ICRAintolerance ใน picsfile

int pics_icra_badexample ถูกกำหนดค่าจากตัวแปร ICRAbadexample ใน picsfile

int pics_icra_pgmaterial ถูกกำหนดค่าจากตัวแปร ICRApgmaterial ใน picsfile

int pics_icra_violencerape ถูกกำหนดค่าจากตัวแปร ICRAviolencerape ใน picsfile

int pics_icra_violencetohumans ถูกกำหนดค่าจากตัวแปร ICRAviolencetohumans ใน picsfile

int pics_icra_violencetoanimals ถูกกำหนดค่าจากตัวแปร ICRAviolencetoanimals ใน picsfile

int pics_icra_violencetofantasy ถูกกำหนดค่าจากตัวแปร ICRAviolencetofantasy ใน picsfile

int pics_icra_violencekillinghumans ถูกกำหนดค่าจากตัวแปร ICRAviolencekillinghumans ใน picsfile

int pics_icra_violencekillinganimals ถูกกำหนดค่าจากตัวแปร ICRAviolencekillinganimals ใน picsfile

int pics_icra_violencekillingfantasy ถูกกำหนดค่าจากตัวแปร ICRAviolencekillingfantasy ใน picsfile

int pics_icra_violenceinjuryhumans ถูกกำหนดค่าจากตัวแปร ICRAviolencejuryhumans ใน picsfile

int pics_icra_violenceinjuryanimals ถูกกำหนดค่าจากตัวแปร ICRAviolencejuryanimals ใน picsfile

int pics_icra_violenceinjuryfantasy ถูกกำหนดค่าจากตัวแปร ICRAviolencejuryfantasy ใน picsfile

int pics_icra_violenceartisitic ถูกกำหนดค่าจากตัวแปร ICRAviolenceartisitc ใน picsfile

int pics_icra_violenceeducational ถูกกำหนดค่าจากตัวแปร ICRAviolenceeducational ใน picsfile

int pics_icra_violencemedical ถูกกำหนดค่าจากตัวแปร ICRAviolencemedical ใน picsfile

int pics_icra_violencesports ถูกกำหนดค่าจากตัวแปร ICRAviolencesports ใน picsfile

int pics_icra_violenceobjects ถูกกำหนดค่าจากตัวแปร ICRAviolenceobjects ใน picsfile

int pics_evaluweb_rating ถูกกำหนดค่าจากตัวแปร evaluWEBrating ใน picsfile

int pics_cybernot_sex ถูกกำหนดค่าจากตัวแปร CyberNOTsex ใน picsfile

int pics_cybernot_other ถูกกำหนดค่าจากตัวแปร CyberNOTother ใน picsfile

int pics_safesurf_agerange ถูกกำหนดค่าจากตัวแปร SafeSurfagerange ใน picsfile

int pics_safesurf_profanity ถูกกำหนดค่าจากตัวแปร SafeSurfprofanity ใน picsfile

int pics_safesurf_heterosexualthemes ถูกกำหนดค่าจากตัวแปร SafeSurfheterosexualthemes ใน picsfile

int pics_safesurf_homosexualthemes ถูกกำหนดค่าจากตัวแปร SafeSurfhomosexualthemes ใน picsfile

int pics_safesurf_nudity ถูกกำหนดค่าจากตัวแปร SafeSurfnudity ใน picsfile

int pics_safesurf_violence ถูกกำหนดค่าจากตัวแปร SafeSurfviolence ใน picsfile

int pics_safesurf_sexviolenceandprofanity ถูกกำหนดค่าจากตัวแปร SafeSurfsexviolenceandprofanity ใน picsfile

int pics_safesurf_intolerance ถูกกำหนดค่าจากตัวแปร SafeSurfintolerance ใน picsfile

int pics_safesurf_druguse ถูกกำหนดค่าจากตัวแปร SafeSurfdruguse ใน picsfile

int pics_safesurf_otheradultthemes ถูกกำหนดค่าจากตัวแปร SafeSurfotheradultthemes ใน picsfile

int pics_safesurf_gambling ถูกกำหนดค่าจากตัวแปร SafeSurfgambling ใน picsfile

int pics_weburbia_rating ถูกกำหนดค่าจากตัวแปร Weburbiarating ใน picsfile

int pics_vancouver_multiculturalism ถูกกำหนดค่าจากตัวแปร Vancouvermulticulturalism ใน picsfile

int pics_vancouver_educationalcontent ถูกกำหนดค่าจากตัวแปร Vancouvereducationalcontent ใน picsfile

int pics_vancouver_environmentalawareness ถูกกำหนดค่าจากตัวแปร Vancouverenvironmentalawareness ใน picsfile

int pics_vancouver_tolerance ถูกกำหนดค่าจากตัวแปร Vancouvertolerance ใน picsfile

int pics_vancouver_violence ถูกกำหนดค่าจากตัวแปร Vancouverviolence ใน picsfile

int pics_vancouver_sex ถูกกำหนดค่าจากตัวแปร Vancouversex ใน picsfile

int pics_vancouver_profanity ถูกกำหนดค่าจากตัวแปร Vancouverprofanity ใน picsfile

int pics_vancouver_safety ถูกกำหนดค่าจากตัวแปร Vancouversafety ใน picsfile

int pics_vancouver_canadiancontent ถูกกำหนดค่าจากตัวแปร Vancouvercanadiancontent ใน picsfile

int pics_vancouver_commercialcontent ถูกกำหนดค่าจากตัวแปร Vancouvercommercialcontent ใน picsfile

int pics_vancouver_gambling ถูกกำหนดค่าจากตัวแปร Vancouvergambling ใน picsfile

std::string magic ถูกกำหนดค่าจากตัวแปร bypasskey ใน filter group configuration file

std::string cookie_magic ตัวแปรสำหรับเก็บค่า bypass cookie magic key

std::string banned_phrase_list_location ถูกกำหนดค่าจากตัวแปร bannedphraselist ใน filter group configuration file

std::string exception_phrase_list_location ถูกกำหนดค่าจากตัวแปร exceptionphraselist ใน filter group configuration file

std::string weighted_phrase_list_location ถูกกำหนดค่าจากตัวแปร weightedphraselist ใน filter group configuration file

std::string banned_site_list_location ถูกกำหนดค่าจากตัวแปร bannedsitelist ใน filter group configuration file

std::string banned_url_list_location ถูกกำหนดค่าจากตัวแปร bannedurllist ใน filter group configuration file

std::string grey_site_list_location ถูกกำหนดค่าจากตัวแปร greysitelist ใน filter group configuration file

std::string grey_url_list_location ถูกกำหนดค่าจากตัวแปร greyurllist ใน filter group configuration file

std::string banned_regexpurl_list_location ถูกกำหนดค่าจากตัวแปร bannedregexpurllist ใน filter group configuration file

std::string content_regexp_list_location ถูกกำหนดค่าจากตัวแปร contentregexplist ใน filter group configuration file

std::string banned_extension_list_location ถูกกำหนดค่าจากตัวแปร bannedextensionlist ใน filter group configuration file

std::string banned_mimetype_list_location ถูกกำหนดค่าจากตัวแปร bannedmimetypelist ใน filter group configuration file

std::string exceptions_site_list_location ถูกกำหนดค่าจากตัวแปร exceptionsitelist ใน filter group configuration file

std::string exceptions_url_list_location ถูกกำหนดค่าจากตัวแปร exceptionurllist ใน filter group configuration file

unsigned int banned_phrase_list เป็นตัวแปรสำหรับเก็บจำนวน banned phrase ที่อยู่ใน list

unsigned int exception_site_list เป็นตัวแปรสำหรับเก็บจำนวน exception site ที่อยู่ใน list

unsigned int exception_url_list เป็นตัวแปรสำหรับเก็บจำนวน exception phrase ที่อยู่ใน list

unsigned int banned_extension_list เป็นตัวแปรสำหรับเก็บจำนวน banned extension ที่อยู่ใน list

unsigned int banned_mimetype_list เป็นตัวแปรสำหรับเก็บจำนวน banned mimetype ที่อยู่ใน list

unsigned int banned_site_list เป็นตัวแปรสำหรับเก็บจำนวน banned site ที่อยู่ใน list

unsigned int banned_url_list เป็นตัวแปรสำหรับเก็บจำนวน banned URL ที่อยู่ใน list

unsigned int grey_site_list เป็นตัวแปรสำหรับเก็บจำนวน grey site ที่อยู่ใน list

unsigned int grey_url_list เป็นตัวแปรสำหรับเก็บจำนวน grey URL ที่อยู่ใน list

unsigned int banned_regexpurl_list เป็นตัวแปรสำหรับเก็บจำนวน banned regular expression URL ที่อยู่ใน list

unsigned int content_regexp_list เป็นตัวแปรสำหรับเก็บจำนวน content regular expression ที่อยู่ใน list

std::deque<RegExp> banned_regexpurl_list_comp เป็น deque สำหรับเก็บ regular expression ของ banned regular expression URL ที่ compile แล้ว

std::deque<String> banned_regexpurl_list_source เป็น deque สำหรับเก็บ string ต้นฉบับ regular expression ของ banned regular expression URL ที่ compile แล้ว

std::deque<RegExp> content_regexp_list_comp เป็น deque สำหรับเก็บ regular expression ของ content regular expression ที่ compile แล้ว

std::deque<String> content_regexp_list_rep เป็น deque สำหรับเก็บ string ที่จะใช้แทนที่ regular expression ของ content regular expression

RegExp pics1 เป็นตัวแปรสำหรับเก็บ regular expression ของ pics-label ที่ compile แล้ว

RegExp pics2 เป็นตัวแปรสำหรับเก็บ regular expression ของ pics rating ที่ compile แล้ว

RegExp isiphost เป็นตัวแปรสำหรับเก็บ regular expression ของ IP host ที่ compile แล้ว

String ada เป็นตัวแปรสำหรับเก็บ access denied address

std::*deque<String> ipToHostname(String ip)* เป็นฟังก์ชั่นที่ใช้เปลี่ยนค่า *ip* ให้กลับมาเป็น domain name โดยค่าที่ return จะเป็น deque ของ domain name ทั้งหมดของ *ip*

ตัวแปรและฟังก์ชั่นแบบ private

std::deque<std::string> conffile เป็น deque สำหรับเก็บตัวแปรต่างๆ ที่กำหนดใน filter group configuration file

bool precompileregexps() เป็นฟังก์ชั่นสำหรับ compile regular expression ที่ต้องใช้เพื่อความรวดเร็วในการทำงาน

bool readbplfile(const char banned, const char* exception, const char* weighted)* เป็นฟังก์ชั่นสำหรับอ่านไฟล์ banned phrase list และกำหนดค่าให้กับตัวแปรต่างๆ

bool readeslfile(const char filename)* เป็นฟังก์ชั่นสำหรับอ่านไฟล์ exception site list ที่กำหนดโดย *filename* และกำหนดค่าให้กับตัวแปรต่างๆ

bool readeurllfile(const char filename)* เป็นฟังก์ชั่นสำหรับอ่านไฟล์ exceptions URL list ที่กำหนดโดย *filename* และกำหนดค่าให้กับตัวแปรต่างๆ

bool readbelfile(const char filename)* เป็นฟังก์ชั่นสำหรับอ่านไฟล์ banned extension list ที่กำหนดโดย *filename* และกำหนดค่าให้กับตัวแปรต่างๆ

bool readbmlfile(const char filename)* เป็นฟังก์ชั่นสำหรับอ่านไฟล์ banned mime type list ที่กำหนดโดย *filename* และกำหนดค่าให้กับตัวแปรต่างๆ

bool readbslfile(const char filename)* เป็นฟังก์ชั่นสำหรับอ่านไฟล์ banned site list ที่กำหนดโดย *filename* และกำหนดค่าให้กับตัวแปรต่างๆ

bool readbulfile(const char filename)* เป็นฟังก์ชั่นสำหรับอ่านไฟล์ banned URL list ที่กำหนดโดย *filename* และกำหนดค่าให้กับตัวแปรต่างๆ

bool readgslfile(const char filename)* เป็นฟังก์ชั่นสำหรับอ่านไฟล์ grey site list ที่กำหนดโดย *filename* และกำหนดค่าให้กับตัวแปรต่างๆ

bool readgulfile(const char filename)* เป็นฟังก์ชั่นสำหรับอ่านไฟล์ grey URL list ที่กำหนดโดย *filename* และกำหนดค่าให้กับตัวแปรต่างๆ

bool readbreulfile(const char filename)* เป็นฟังก์ชั่นสำหรับอ่านไฟล์ banned regular expression URL list ที่กำหนดโดย *filename* และกำหนดค่าให้กับตัวแปรต่างๆ

bool compilebreulfile(unsigned int list) เป็นฟังก์ชั่นสำหรับ compile banned regular expression URL list และกำหนดค่าให้กับตัวแปรต่างๆ

bool readcrelfile(const char filename)* เป็นฟังก์ชั่นสำหรับอ่านไฟล์ content regular expression list ที่กำหนดโดย *filename* และกำหนดค่าให้กับตัวแปรต่างๆ

int findoptionI(const char option)* เป็นฟังก์ชั่นสำหรับอ่านค่าตัวแปร *option* ที่เป็น integer จาก deque *conffile*

std::string findoptionS(const char option)* เป็นฟังก์ชั่นสำหรับอ่านค่าตัวแปร *option* ที่เป็น string จาก deque *conffile*

bool realitycheck(String s, int minl, int maxl, char emessage)* เป็นฟังก์ชั่นสำหรับตรวจสอบค่าของ *s* ว่ามีความยาวอยู่ในช่วงที่ควรจะเป็นหรือไม่

2.3.5 Socket

เป็น class สำหรับเก็บและจัดการกับ Internet socket โดยมีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ตัวแปรและฟังก์ชั่นแบบ public

Socket() เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า default

~Socket() เป็น destructor ของ class นี้ ใช้ลบค่าต่างๆ จากหน่วยความจำเพื่อป้องกันปัญหา memory leak

Socket(int newfd, struct sockaddr_in myip, struct sockaddr_in peerip) เป็น constructor ของ class นี้ที่กำหนดค่าเริ่มต้นตาม *newfd, myip, peerip*

void reset() เป็นฟังก์ชั่นที่ใช้กำหนดค่าเริ่มต้นใหม่ให้กับ socket

int connect(std::string ip, int port) เป็นฟังก์ชั่นที่ใช้สร้าง connection ไปยัง *ip* และ *port* ที่กำหนด

int bind(int port) เป็นฟังก์ชั่นที่ใช้ bind socket เข้ากับ *port* ที่กำหนด

int bind(std::string ip, int port) เป็นฟังก์ชั่นที่ใช้ bind socket เข้ากับ *ip* และ *port* ที่กำหนด

int listen(int queue) เป็นฟังก์ชั่นที่ใช้กำหนดให้ socket รอรับ connection โดยที่มีคิวขนาด *queue* requests

Socket accept() เป็นฟังก์ชั่นที่ใช้รับ connection

std::string getPeerIP() เป็นฟังก์ชั่นที่ return IP addres ของ client ที่กำลังติดต่ออยู่

int getPeerSourcePort() เป็นฟังก์ชั่นที่ return ยนพะ ของ client ที่กำลังติดต่ออยู่

int getFD() เป็นฟังก์ชั่นที่ return descriptor ของ socket

void setFD(int newfd) เป็นฟังก์ชั่นที่ใช้กำหนดค่า descriptor ของ socket ตาม *newfd*

void close() เป็นฟังก์ชั่นที่ใช้ปิด socket

void setTimeout(int t) เป็นฟังก์ชั่นที่ใช้กำหนด timeout ของ socket ตาม *t*

int getTimeout() เป็นฟังก์ชั่นที่ return ค่า timeout ของ socket

bool checkForInput() เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่ามีข้อมูลเข้ามาที่ socket หรือไม่โดยไม่มีการรอ

void checkForInput(int timeout) throw(exception) เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่ามีข้อมูลเข้ามาที่ socket หรือไม่โดยจะรอเป็นเวลา *timeout* วินาทีและ throw exception เมื่อเกิดข้อผิดพลาด

bool readyForOutput() เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า socket พร้อมที่จะส่งข้อมูลออกไปหรือไม่โดยไม่มีการรอ

void readyForOutput(int timeout) throw(exception) เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า socket พร้อมที่จะส่งข้อมูลออกไป หรือไม่โดยจะรอเป็นเวลา *timeout* วินาที และ throw exception เมื่อเกิดข้อผิดพลาด

int getline(char buff, int size, int timeout) throw(exception)* เป็นฟังก์ชั่นที่ใช้อ่านค่าจาก socket เข้ามาใส่ที่ตัวแปร *buff* โดยตัวแปร *buff* มีขนาด *size* และจะรอเป็นเวลา *timeout* ถ้าเกิดข้อผิดพลาดก็จะ throw exception

void writeString(const char line) throw(exception)* เป็นฟังก์ชั่นที่ใช้เขียนค่า *line* ออกไปยัง UDsocket และ throw exception เมื่อเกิดข้อผิดพลาด

bool writeToSocket(char buff, int len, unsigned int flags, int timeout)* เป็นฟังก์ชั่นที่ใช้เขียนค่า *buff* ที่มีความยาว *len* ออกไปยัง UDsocket โดยจะรอให้ UDsocket พร้อมเป็นเวลา *timeout* วินาที

int readFromSocketn(char buff, int len, unsigned int flags, int timeout)* เป็นฟังก์ชั่นที่ใช้อ่านค่าจาก UDsocket เข้ามาใส่ *buff* เป็นจำนวน *len* ไบต์โดยจะรอให้ UDsocket พร้อมเป็นเวลา *timeout* วินาที และ return ค่าเป็นจำนวนไบต์ที่อ่านได้จริงๆ

int readFromSocket(char buff, int len, unsigned int flags, int timeout)* เป็นฟังก์ชั่นที่ใช้อ่านค่าทั้งหมดจาก UDsocket เข้ามาใส่ *buff* โดยที่ *buff* มีขนาด *len* ไบต์ และจะรอให้ UDsocket พร้อมเป็นเวลา *timeout* วินาที ค่าที่ return จะเป็นจำนวนไบต์ที่อ่านได้

void writeToSockete(char buff, int len, unsigned int flags, int timeout) throw(exception)* เป็นฟังก์ชั่นที่ใช้เขียนค่า *buff* ที่มีความยาว *len* ออกไปยัง UDsocket โดยจะรอให้ UDsocket พร้อมเป็นเวลา *timeout* วินาที และ throw exception เมื่อเกิดข้อผิดพลาด

ตัวแปรและฟังก์ชั่นแบบ private

int timeout เป็นตัวแปรสำหรับเก็บค่า timeout ของ socket

bool isclosed เป็นตัวแปรสำหรับบอกว่าตอนนี้ socket ปิดอยู่หรือไม่

bool isused เป็นตัวแปรสำหรับบอกว่าตอนนี้ socket ถูกใช้งานอยู่หรือไม่

socklen_t peer_adr_in_length เป็นตัวแปรสำหรับเก็บความยาวของ address ที่ได้รับกลับมาจากการเรียก system call acccept

int sck_inet เป็นตัวแปรสำหรับเก็บค่า descriptor สำหรับอ้างถึง socket

struct sockaddr_in my_adr_inet เป็นตัวแปรสำหรับเก็บค่า structure ของ socket สำหรับเครื่องที่ทำงานอยู่

struct sockaddr_in peer_adr_inet เป็นตัวแปรสำหรับเก็บค่า structure ของ socket สำหรับเครื่อง client

*int selectEINTR(int numfds, fd_set * readfds, fd_set * writefds, fd_set * exceptfds, struct timeval * timeout)* เป็นฟังก์ชั่นที่ใช้เรียกฟังก์ชั่น select โดยจะมีการเรียกซ้ำถ้าได้รับ signal

2.3.6 UDSocket

เป็น class สำหรับเก็บและจัดการกับ Unix Datagram socket โดยมีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ตัวแปรและฟังก์ชั่นแบบ public

UDSocket() เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า default

~Socket() เป็น destructor ของ class นี้ ใช้ลบค่าต่างๆ จากหน่วยความจำเพื่อป้องกันปัญหา memory leak

UDSocket(int newfd, struct sockaddr_un myadr) เป็น constructor ของ class นี้ที่กำหนดค่าเริ่มต้นตาม *newfd, myadr*

void reset() เป็นฟังก์ชั่นที่ใช้กำหนดค่าเริ่มต้นใหม่ให้กับ UDsocket

int connect(const char path)* เป็นฟังก์ชั่นที่ใช้สร้าง connection เข้ากับ *path* ที่กำหนด

int bind(const char path)* เป็นฟังก์ชั่นที่ใช้ bind UDsocket เข้ากับ *path* ที่กำหนด

int listen(int queue) เป็นฟังก์ชั่นที่ใช้กำหนดให้ UDsocket รอรับ connection โดยที่มีคิวขนาด *queue* requests

UDSocket accept() เป็นฟังก์ชั่นที่ใช้รับ connection

int getFD() เป็นฟังก์ชั่นที่ return descriptor ของ UDsocket

void setFD(int newfd) เป็นฟังก์ชั่นที่ใช้กำหนดค่า descriptor ของ UDsocket ตาม *newfd*

void close() เป็นฟังก์ชั่นที่ใช้ปิด UDsocket

void setTimeout(int t) เป็นฟังก์ชั่นที่ใช้กำหนด timeout ของ UDsocket ตาม *t*

int getTimeout() เป็นฟังก์ชั่นที่ return ค่า timeout ของ UDsocket

bool checkForInput() เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่ามีข้อมูลเข้ามาที่ UDsocket หรือไม่โดยไม่มีการรอ

void checkForInput(int timeout) throw(exception) เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่ามีข้อมูลเข้ามาที่ UDsocket หรือไม่โดยจะรอเป็นเวลา *timeout* วินาทีและ throw exception เมื่อเกิดข้อผิดพลาด

bool readyForOutput() เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า UDsocket พร้อมที่จะส่งข้อมูลออกไป หรือไม่โดยไม่มีการรอ

void readyForOutput(int timeout) throw(exception) เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า UDsocket พร้อมที่จะส่งข้อมูลออกไป หรือไม่โดยจะรอเป็นเวลา *timeout* วินาที และ throw exception เมื่อเกิดข้อผิดพลาด

int getline(char buff, int size, int timeout) throw(exception)* เป็นฟังก์ชั่นที่ใช้อ่านค่าจาก UDsocket เข้ามาใส่ที่ตัวแปร *buff* โดยตัวแปร *buff* มีขนาด *size* และจะรอเป็นเวลา *timeout* ถ้าเกิดข้อผิดพลาดก็จะ throw exception

void writeString(const char line) throw(exception)* เป็นฟังก์ชั่นที่ใช้เขียนค่า *line* ออกไปยัง socket และ throw exception เมื่อเกิดข้อผิดพลาด

bool writeToSocket(char buff, int len, unsigned int flags, int timeout)* เป็นฟังก์ชั่นที่ใช้เขียนค่า *buff* ที่มีความยาว *len* ออกไปยัง socket โดยจะรอให้ socket พร้อมเป็นเวลา *timeout* วินาที

int readFromSocketn(char buff, int len, unsigned int flags, int timeout)* เป็นฟังก์ชั่นที่ใช้อ่านค่าจาก socket เข้ามาใส่ *buff* เป็นจำนวน *len* ไบต์โดยจะรอให้ socket พร้อมเป็นเวลา *timeout* วินาที และ return ค่าเป็นจำนวนไบต์ที่อ่านได้จริงๆ

int readFromSocket(char buff, int len, unsigned int flags, int timeout)* เป็นฟังก์ชั่นที่ใช้อ่านค่าทั้งหมดจาก socket เข้ามาใส่ *buff* โดยที่ *buff* มีขนาด *len* ไบต์ และจะรอให้ socket พร้อมเป็นเวลา *timeout* วินาที ค่าที่ return จะเป็นจำนวนไบต์ที่อ่านได้

void writeToSockete(char buff, int len, unsigned int flags, int timeout) throw(exception)* เป็นฟังก์ชั่นที่ใช้เขียนค่า *buff* ที่มีความยาว *len* ออกไปยัง socket โดยจะรอให้ socket พร้อมเป็นเวลา *timeout* วินาที และ throw exception เมื่อเกิดข้อผิดพลาด

ตัวแปรและฟังก์ชั่นแบบ private

int timeout เป็นตัวแปรสำหรับเก็บค่า timeout ของ UDsocket

bool isclosed เป็นตัวแปรสำหรับบอกว่าตอนนี้ UDsocket ปิดอยู่หรือไม่

bool isused เป็นตัวแปรสำหรับบอกว่าตอนนี้ UDsocket ถูกใช้งานอยู่หรือไม่

socklen_t peer_adr_un_length เป็นตัวแปรสำหรับเก็บความยาวของ address ที่ได้รับกลับมาจากการเรียก system call acccept

int sck_un เป็นตัวแปรสำหรับเก็บค่า descriptor สำหรับอ้างถึง UDsocket

struct sockaddr_un my_adr_un เป็นตัวแปรสำหรับเก็บค่า structure ของ UDsocket สำหรับผู้เรียกผ่าน UDSocket

struct sockaddr_un peer_adr_un เป็นตัวแปรสำหรับเก็บค่า structure ของ UDsocket สำหรับผู้ถูกเรียกผ่าน UDSocket

*int selectEINTR(int numfds, fd_set * readfds, fd_set * writefds, fd_set * exceptfds, struct timeval * timeout)* เป็นฟังก์ชั่นที่ใช้เรียกฟังก์ชั่น select โดยจะมีการเรียกซ้ำถ้าได้รับ signal

2.3.7 ListManager

เป็น class สำหรับจัดการกับ deque (linked list) ที่ใช้เก็บค่า content filtering ต่างๆ จาก filter group configuration files โดยมีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ตัวแปรและฟังก์ชั่นแบบ public

~ListManager() เป็น destructor ของ class นี้ ใช้ลบค่าต่างๆ จากหน่วยความจำเพื่อป้องกันปัญหา memory leak

*int newItemList(const char *filename, bool startswith, int filters, bool parent)* เป็นฟังก์ชั่นสำหรับสร้าง deque (linked list) ขึ้นมาใหม่จากไฟล์ *filename* และนำไปใส่ใน deque *l*

*int newPhraseList(const char *exception, const char *banned, const char *weighted)* เป็นฟังก์ชั่นสำหรับเพิ่มค่าของ *exception, banned* และ *weighted* เข้าไปใน deque *l*

void deRefList(unsigned int item) เป็นฟังก์ชั่นที่ใช้ลดค่าของ *refcount* ใน deque *l* ที่ตำแหน่ง *item*

void garbageCollect() เป็นฟังก์ชั่นที่ใช้ลบค่าใน deque *l* เมื่อไม่ได้ใช้งานแล้ว

std::deque<ListContainer> l* เป็นตัวแปรที่ใช้เก็บ deque ที่เก็บค่า content filtering ต่างๆ ที่กำหนดจาก filter group configuration file

ฟังก์ชั่นแบบ private

int findNULL() เป็นฟังก์ชั่นที่ใช้สำหรับหาตำแหน่งของใน deque *l* ที่ยังว่างอยู่

int getFileDate(const char filename)* เป็นฟังก์ชั่นที่ใช้หาเวลาที่มีการแก้ไข *filename* ครั้งล่าสุด โดย return ค่าเป็นจำนวนวินาทีจากเวลา UTC 00:00.00 ของวันที่ 1 มกราคม ปี 1970

2.3.8 ListContainer

เป็น class สำหรับเก็บค่า content filtering ต่างๆ จาก filter group configuration files ซึ่งจะถูกสร้างและเก็บอยู่ภายใน class ListManager โดยมีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ตัวแปรและฟังก์ชั่นแบบ public

ListContainer() เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า default

~ListContainer() เป็น destructor ของ class นี้ ใช้ลบค่าต่างๆ จากหน่วยความจำเพื่อป้องกันปัญหา memory leak

void reset() เป็นฟังก์ชั่นที่ใช้ reset ค่าตัวแปรต่างๆ ของ object

bool readPhraseList(const char filename, bool isexception)* เป็นฟังก์ชั่นที่ใช้อ่านค่าวลีจาก *filename* และเก็บค่าไว้ในตัวแปรที่เหมาะสม

bool readItemList(const char filename, bool startswith, int filters)* เป็นฟังก์ชั่นที่ใช้อ่านค่าจาก *filename* และเก็บไว้ในตัวแปรที่เหมาะสม

bool inList(char string)* เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า *string* อยู่ใน list หรือไม่โดยใช้ binary search

bool inListEndsWith(char string)* เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่าค่ามีใน list ที่ลงท้ายด้วย *string* หรือไม่โดยใช้ binary search

bool inListStartsWith(char string)* เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่ามีค่าใน list ที่ขึ้นต้นด้วย *string* หรือไม่โดยใช้ binary search

char findInList(char* string)* เป็นฟังก์ชั่นที่ใช้ค้าหาว่ามี *string* อยู่ใน list หรือไม่โดยใช้ binary search และ return ค่าเป็น pointer ไปยังค่าที่หาพบ หรือ NULL ถ้าไม่พบ

char findEndsWith(char* string)* เป็นฟังก์ชั่นที่ใช้ค้าหาว่ามีค่าใน list ที่ลงท้ายด้วย *string* อยู่หรือไม่โดยใช้ binary search และ return ค่าเป็น pointer ไปยังค่าที่หาพบ หรือ NULL ถ้าไม่พบ

char findStartsWith(char* string)* เป็นฟังก์ชั่นที่ใช้ค้าหาว่ามีค่าใน list ที่ขึ้นต้นด้วย *string* อยู่หรือไม่โดยใช้ binary search และ return ค่าเป็น pointer ไปยังค่าที่หาพบ หรือ NULL ถ้าไม่พบ

char findStartsWithPartial(char* string)* เป็นฟังก์ชั่นที่ใช้ค้าหาว่ามีค่าใน list ที่ขึ้นต้นด้วย *string* อยู่หรือไม่โดยใช้ binary search และ return ค่าเป็น pointer ไปยังค่าที่หาพบ หรือค่าที่ใกล้เคียง

int getListLength() เป็นฟังก์ชั่นที่ใช้ return ค่า *items* (จำนวน items ใน list)

std::string getItemAt(char index)* เป็นฟังก์ชั่นที่ใช้ return ค่า string จากค่าของ *index*

std::string getItemAtInt(int index) เป็นฟังก์ชั่นที่ใช้ return ค่า string ตัวที่ *index* จากตัวแปร *data*

int getWeightAt(unsigned int index) เป็นฟังก์ชั่นที่ใช้ return ค่า weight ที่ตำแหน่ง *index*

int getTypeAt(unsigned int index) เป็นฟังก์ชั่นที่ใช้ return ค่า itemtype ที่ตำแหน่ง *index*

void endsWithSort() เป็นฟังก์ชั่นที่ใช้เรียงลำดับข้อมูลโดยเรียงจากท้ายบรรทัดโดยใช้ quick sort

void startsWithSort() เป็นฟังก์ชั่นที่ใช้เรียงลำดับข้อมูลโดยเรียงจากต้นบรรทัดโดยใช้ quick sort

bool createCacheFile() เป็นฟังก์ชั่นที่ใช้สร้าง cache file ของข้อมูลใน list เพื่อเวลาเรียกใช้งานในครั้งต่อไปจะได้ไม่ต้องเสียเวลาสร้าง list ใหม่

void makeGraph(int fqs) เป็นฟังก์ชั่นที่ใช้สร้าง graph สำหรับค้นหาข้อมูลโดยถ้า *fqs* (force quick search) มีค่าเป็น 1 จะใช้ quick search ถ้า *fqs* มีค่าเป็น 0 จะใช้ DFA (Deterministic Finite Automaton) search algorithm

bool previousUseItem(const char filename, bool startswith, int filters)* เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า *filename* เคยถูกนำมาใช้งานแล้วหรือไม่

bool upToDate() เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่าข้อมูลต่างๆ ยังทันสมัยอยู่หรือไม่ จะ return ค่าเป็น false ถ้า configureation files ต่างๆ ได้ถูกแก้ไข

std::deque<unsigned int> graphSearch(char doc, int len)* เป็นฟังก์ชั่นที่ใช้ค้าหาว่าใน *doc* ที่มีความยาว *len* มีวลีต่างๆ ที่อยู่ในกราฟจำนวนเท่าไร และ return ค่าเป็น deque ของตำแหน่งของวลีที่พบในกราฟ

std::deque<int> combilist เป็นตัวแปรที่ใช้เก็บค่า combination list

int refcount เป็นตัวแปรที่ใช้นับว่ามีการอ้างถึง list นี้กี่ครั้ง

bool parent เป็นตัวแปรที่ใช้บอกว่า list นี้เป็น parent list หรือไม่

int filedate เป็นตัวแปรที่ใช้เก็บค่าเวลาที่มีการแก้ไขไฟล์ครั้งล่าสุด โดยจะมีค่าเป็นจำนวนวินาทีจากเวลา UTC 00:00.00 ของวันที่ 1 มกราคม ปี 1970

bool used เป็นตัวแปรที่ใช้บอกว่า list นี้ถูกใช้งานหรือยัง

String bannedpfile เป็นตัวแปรที่ใช้เก็บชื่อไฟล์ของ banned phrase list

String exceptionpfile เป็นตัวแปรที่ใช้เก็บชื่อไฟล์ของ exception phrase list

String weightedpfile เป็นตัวแปรที่ใช้เก็บชื่อไฟล์ของ weighted phrase list

int bannedpfiledate เป็นตัวแปรที่ใช้เก็บเวลาที่แก้ไขไฟล์ banned phrase list ครั้งล่าสุดโดยจะมีค่าเป็นจำนวนวินาทีจากเวลา UTC 00:00.00 ของวันที่ 1 มกราคม ปี 1970

int exceptionpfiledate เป็นตัวแปรที่ใช้เก็บเวลาที่แก้ไขไฟล์ exception phrase list ครั้งล่าสุดโดยจะมีค่าเป็นจำนวนวินาทีจากเวลา UTC 00:00.00 ของวันที่ 1 มกราคม ปี 1970

int weightedpfiledate เป็นตัวแปรที่ใช้เก็บเวลาที่แก้ไขไฟล์ weighted phrase list ครั้งล่าสุดโดยจะมีค่าเป็นจำนวนวินาทีจากเวลา UTC 00:00.00 ของวันที่ 1 มกราคม ปี 1970

String sourcefile เป็นตัวแปรที่ใช้เก็บชื่อไฟล์ที่ใช้สร้าง list

std::deque<unsigned int> morelists เป็นตัวแปรที่ใช้เก็บจำนวนวลีของแต่ละ child list

ตัวแปรและฟังก์ชั่นแบบ private

bool sourceisexception เป็นตัวแปรที่ใช้เก็บว่าเป็น exception phrase list หรือไม่

bool sourcestartswith เป็นตัวแปรที่ใช้เก็บค่า startswith ของ source file

int sourcefilters เป็นตัวแปรที่ใช้เก็บค่า filter ของ sourcefile

char data* เป็นตัวแปรที่ใช้เก็บวลีโดยแต่ละวลีจะแยกจากกันด้วย '\0'

int graphdata* เป็นตัวแปรที่ใช้เก็บ DFA search graph

int graphitems เป็นตัวแปรที่ใช้เก็บจำนวน item ใน search graph

std::deque<unsigned int> slowgraph เป็นตัวแปรที่ใช้เก็บ quick search graph

unsigned int data_length เป็นตัวแปรที่ใช้บอกว่า *data* ที่มีข้อมูลเก็บอยู่กี่ไบต์

unsigned int data_memory เป็นตัวแปรที่ใช้เก็บขนาดของ *data*

int items เป็นตัวแปรที่ใช้เก็บว่ามีวลีอยู่กี่วลี ใน *data*

bool isSW เป็นตัวแปรที่ใช้บอกว่า list นี้เรียงลำดับโดยต้นบรรทัด (true) หรือท้ายบรรทัด (false)

bool issorted เป็นตัวแปรที่ใช้บอกว่า list นี้เรียงลำดับแล้วหรือยัง

bool graphused เป็นตัวแปรที่ใช้บอกว่าสร้าง graph หรือยัง

std::deque<unsigned int> list เป็นตัวแปรที่ใช้เก็บตำแหน่งของวลีว่าอยู่ที่ไหนใน *data*

std::deque<int> weight เป็นตัวแปรที่ใช้เก็บน้ำหนักของวลี

std::deque<int> itemtype เป็นตัวแปรที่ใช้เก็บชนิดของวลี โดยค่า 0 คือ banned phrase, 1 คือ weighted phrase และ -1 คือ exception phrase

int force_quick_search เป็นตัวแปรที่ใช้กำหนดว่าจะ search graph ด้วย quick search หรือ DFA search algorithm

bool readAnotherItemList(const char filename, bool startswith, int filters)* เป็นฟังก์ชั่นที่ใช้อ่าน *filename* และกำหนดค่าให้กับตัวแปรที่เหมาะสม จะถูกเรียกใช้เมื่อใน configuration file มีการ include ไฟล์อื่นๆ เข้ามา

void readPhraseListHelper(String line, bool isexception) เป็นฟังก์ชั่นที่ใช้อ่านค่าวลีจาก *line* และกำหนดค่าโดยเรียกใช้ *readPhraseListHelper2*

void readPhraseListHelper2(String phrase, int type, int weighting) เป็นฟังก์ชั่นที่ใช้กำหนดค่า *phrase, type* และ *weighting* โดยเรียกใช้ *addToItemListPhrase*

bool addToItemListPhrase(char s, int len, int type, int weighting, bool combi)* เป็นฟังก์ชั่นที่ใช้กำหนดค่าให้กับตัวแปร *data, combilist, weight* และ *itemtype* ใน class นี้

void graphSizeSort(int l, int r, std::deque<unsigned int> sizelist)* เป็นฟังก์ชั่นที่ใช้เรียงลำดับข้อมูลใน DFA search graph ตามขนาด

void graphAdd(String s, int inx, int item) เป็นฟังก์ชั่นที่ใช้เพิ่มข้อมูลเข้าไปใน DFA search graph

int graphFindBranches(unsigned int pos) เป็นฟังก์ชั่นที่ใช้หาจำนวนของ branches ที่ node *pos* มี

void graphCopyNodePhrases(unsigned int pos) เป็นฟังก์ชั่นที่ใช้ก๊อปปี้ค่าจาก DFA search graph ไปยัง quick search graph

int bmsearch(char file, int fl, std::string s)* เป็นฟังก์ชั่นที่ใช้ค้นหาว่า *file* ที่มีขนาด *fl* มี string *s* อยู่ภายในหรือไม่ return ค่าเป็นจำนวน string *s* ที่พบใน *file* การค้นหาทำโดยใช้ Boyer Moore quick search algorithm

void quicksortSW(int l, int r) เป็นฟังก์ชั่นที่ใช้ทำ quick sort ข้อมูลใน *data* โดยเรียงลำดับด้วยต้นบรรทัด

void quicksortEW(int l, int r) เป็นฟังก์ชั่นที่ใช้ทำ quick sort ข้อมูลใน *data* โดยเรียงลำดับด้วยท้ายบรรทัด

bool readProcessedItemList(const char filename, bool startswith, int filters)* เป็นฟังก์ชั่นที่ใช้อ่านค่าจาก cache file *filename*

void addToItemList(char s, int len)* เป็นฟังก์ชั่นที่ใช้เพิ่มข้อมูล *s* ที่มีขนาด *len* เข้าไปใน list

int greaterThanEWF(const char a, const char* b)* เป็นฟังก์ชั่นที่ใช้เปรียบเทียบว่า *a* มากกว่า *b* หรือไม่แบบ full match โดยเทียบจากด้านท้าย

int greaterThanEW(const char a, const char* b)* เป็นฟังก์ชั่นที่ใช้เปรียบเทียบว่า *a* มากกว่า *b* หรือไม่แบบ partial match โดยเทียบจากด้านท้าย

int greaterThanSWF(const char a, const char* b)* เป็นฟังก์ชั่นที่ใช้เปรียบเทียบว่า *a* มากกว่า *b* หรือไม่แบบ full match โดยเทียบจากด้านหน้า

int greaterThanSW(const char a, const char* b)* เป็นฟังก์ชั่นที่ใช้เปรียบเทียบว่า *a* มากกว่า *b* หรือไม่แบบ partial match โดยเทียบจากด้านหน้า

int searchREWF(int a, int s, const char p)* เป็นฟังก์ชั่นที่ใช้ค้นหาว่า *p* อยู่ใน *data* ในช่วงระหว่าง *a* ถึง *s* หรือไม่ โดยเปรียบเทียบจากด้านท้าย แบบ full match

int searchREW(int a, int s, const char p)* เป็นฟังก์ชั่นที่ใช้ค้นหาว่า *p* อยู่ใน *data* ในช่วงระหว่าง *a* ถึง *s* หรือไม่ โดยเปรียบเทียบจากด้านท้าย แบบ partial match

int searchRSW(int a, int s, const char p)* เป็นฟังก์ชั่นที่ใช้ค้นหาว่า *p* อยู่ใน *data* ในช่วงระหว่าง *a* ถึง *s* หรือไม่ โดยเปรียบเทียบจากด้านหน้า แบบ partial match

int searchRSWF(int a, int s, const char* p) เป็นฟังก์ชั่นที่ใช้ค้นหาว่า *p* อยู่ใน *data* ในช่วงระหว่าง *a* ถึง *s* หรือไม่ โดยเปรียบเทียบจากด้านหน้า แบบ full match

bool isCacheFileNewer(const char string)* เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า cache file (*string*.process) ใหม่กว่า ไฟล์ต้นฉบับ (*string*)หรือไม่

int getFileLength(const char filename)* เป็นฟังก์ชั่นที่ใช้หาขนาดของไฟล์ *filename*

int getFileDate(const char filename)* เป็นฟังก์ชั่นที่ใช้หาเวลาที่แก้ไขไฟล์ *filename* ครั้งล่าสุดโดยจะมีค่าเป็นจำนวนวินาทีจากเวลา UTC 00:00.00 ของวันที่ 1 มกราคม ปี 1970

void increaseMemoryBy(int bytes) เป็นฟังก์ชั่นที่ใช้เพิ่มขนาดของ *data* ไปอีกเป็นจำนวน *byte*

std::string toLower(std::string s) เป็นฟังก์ชั่นที่ใช้เปลี่ยนค่า *s* ให้เป็นตัวอักษรตัวเล็กทั้งหมด

2.3.9 SysV

เป็น class สำหรับจัดการกับ process ของในแบบ SystemV โดยมีฟังก์ชั่นต่างๆ ดังต่อไปนี้

ฟังก์ชั่นแบบ public

int kill(std::string pidfile) เป็นฟังก์ชั่นที่ใช้ kill process ที่มี pid อยู่ใน *pidfile*

int showPID(std::string pidfile) เป็นฟังก์ชั่นที่ใช้แสดงค่า pid ที่อยู่ใน *pidfile* ออกมาที่ standard output file (หน้าจอ)

int openPIDFile(std::string pidfile) เป็นฟังก์ชั่นที่ใช้สร้างไฟล์ *pidfile* ขึ้นมาเพื่อใช้เก็บ pid ของโปรแกรมที่ทำงานอยู่

int writePIDFile(int pidfilefd) เป็นฟังก์ชั่นที่ใช้เขียน pid ของโปรแกรมเข้าไปในไฟล์ที่กำหนดโดย descriptor *pidfilefd*

bool amIRunning(std::string pidfile) เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่าโปรแกรมทำงานอยู่หรือไม่ โดยตรวจดูว่ามีไฟล์ *pidfile* อยู่หรือไม่ถ้ามีแสดงว่าโปรแกรมทำงานอยู่

int hup(std::string pidfile) เป็นฟังก์ชั่นที่ใช้ส่ง signal SIGHUP ไปให้โปรแกรมที่มี pid ตามค่าที่อยู่ใน *pidfile*

int usr1(std::string pidfile) เป็นฟังก์ชั่นที่ใช้ส่ง signal SIGUSR1 ไปให้โปรแกรมที่มี pid ตามค่าที่อยู่ใน *pidfile*

ฟังก์ชั่นแบบ private

pid_t getPID(std::string pidfile) เป็นฟังก์ชั่นที่ใช้อ่านค่า pid จาก *pidfile* หรือจาก command

pid_t getPIDFromFile(std::string pidfile) เป็นฟังก์ชั่นที่ใช้อ่านค่า pid จาก *pidfile*

pid_t getPIDFromCommand(const char command)* เป็นฟังก์ชั่นที่ใช้อ่านค่า pid จาก *command*

bool confirmName(pid_t p) เป็นฟังก์ชั่นที่ใช้ทดสอบว่า *p* เป็น pid ของโปรแกรมเราจริงหรือไม่

2.3.10 ConnectionHandler

เป็น class สำหรับจัดการกับการติดต่อกับ client ซึ่งทำงานอยู่ภายใน worker process ในรูปที่ 2.6 โดยมีฟังก์ชั่นต่างๆ ดังต่อไปนี้

ฟังก์ชั่นแบบ public

void handleConnection(int peerfd, String ip, int port) เป็นฟังก์ชั่นที่ใช้จัดการในการติดต่อกับ client ผ่านทาง descriptor *peerfd*, *ip* และ *port* ที่กำหนดโดยมีรายละเอียดการทำงานดังนี้

1) กำหนดค่าเริ่มต้นให้กับตัวแปรต่างๆ
2) สร้างการเชื่อมต่อไปยังพรอกซี่เซิร์ฟเวอร์
3) รับข้อมูลจาก client และตรวจสอบดูว่าเป็น URL ที่ถูกต้องหรือไม่ ถ้าไม่ถูกต้องก็จะส่งข้อความเตือนกลับไปยัง client
4) ตรวจสอบว่า user ที่เข้ามาจาก client อยู่ใน filter group ไหน ถ้าไม่กำหนดก็จะอยู่ใน default group

5) ตรวจสอบว่ามีการ bypass URL หรือ bypass cookie หรือไม่และกำหนดตัวแปรให้ถูกต้อง
6) ตรวจสอบว่าอยู่ใน exception lists ต่างๆหรือไม่
7) หากอยู่ใน exception lists และไม่อยู่ใน banned IP list และ banned user list ก็ทำการ tunnel connection จาก client ไปให้พรอกซี่เซิร์ฟเวอร์โดยตรงและออกจากการทำงานเมื่อ client โอนถ่ายข้อมูลเสร็จ
8) ถ้าเป็นการเชื่อมต่อแบบ SSL ก็จะตรวจเฉพาะ HTTP header เท่านั้นเนื่องจากไม่สามารถถอดรหัสได้ เมื่อตรวจผ่านก็จะ tunnel connection จาก client ไปให้พรอกซี่เซิร์ฟเวอร์โดยตรงและออกจากการทำงานเมื่อ client โอนถ่ายข้อมูลเสร็จ
9) ส่ง request ไปให้พรอกซี่เซิร์ฟเวอร์และรอรับ HTTP reply header กลับมาและตรวจสอบว่าอยู่ใน banned MIMT type list หรือไม่
10) ตรวจดูว่าเป็น redirection หรือไม่ ถ้าเป็นก็ตรวจดู disposition file name ว่าอยู่ใน banned extension list หรือไม่
11) ตรวจสอบ request นี้โดยเรียก *requestChecks()*
12) รับ HTML body จากพรอกซี่เซิร์ฟเวอร์ถ้ามีการ compress ก็ทำการ decompress เสียก่อน จากนั้นจึงทำการตรวจดูว่าหน้าเว็บนี้เคยผ่านการตรวจสอบแล้วหรือยัง ถ้าเคยผ่านมาแล้วก็ไม่จำเป็นต้องตรวจสอบอีก แต่ถ้ายังไม่เคยก็ทำการตรวจสอบด้วยฟังก์ชั่น *checkme()* ของ class NaughtyFilter
13) ถ้ามีการกำหนด content modified regular expressions ก็ทำการแทนค่า HTML body ด้วย regular expressions
14) ถ้าตรวจ content filtering ผ่านและยังไม่เคยเก็บค่า URL นี้ไว้ใน cache file ก็ทำการเขียน URL นี้เก็บใน cache file ด้วย เพื่อการเรียกครั้งต่อไปจะได้ไม่ต้องตรวจสอบอีก
15) ถ้าไม่ผ่านการตรวจก็จะเขียนข้อมูลลงใน log file และแสดงข้อความเตือนกลับมายัง client โดยเรียกใช้ฟังก์ชั่น *denyAccess()* แล้วออกจากการทำงาน
16) เมื่อผ่านการตรวจทั้งหมดแล้วก็ส่ง HTML body ต่อไปให้ client แล้วปิด socket และออกจากการทำงาน

ฟังก์ชั่นแบบ private

*void doTheLogMan(std::string who, std::string from, std::string where, std::string what, std::string how, int size, struct timeval *thestart, bool cachehit, int code, std::string mimetype)* เป็นฟังก์ชั่นที่ใช้เขียนข้อมูลลงใน logfile

std::string miniURLEncode(std::string s) เป็นฟังก์ชั่นที่ใช้เปลี่ยนตัวอักษรใน *s* ที่ไม่ใช่ภาษาอังกฤษหรือตัวเลขให้เป็นเลขฐาน 16 ที่ขึ้นต้นด้วย %

*void decideHowToLog(std::string who, std::string from, std::string where, unsigned int port, std::string what, std::string how, int size, int loglevel, bool isnaughtly, bool isexception, int logexceptions, bool istext, struct timeval *thestart, bool cachehit, int code, std::string mimetype)* เป็นฟังก์ชั่นที่ใช้เตรียมข้อมูลเพื่อเขียนลงใน log file โดยการเรียกใช้ฟังก์ชั่น *doTheLogMan()*

bool wasClean(String url) เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า *url* อยู่ใน URL list cache หรือไม่

void addToClean(String url) เป็นฟังก์ชั่นที่ใช้เพิ่ม *url* เข้าไปใน URL list cache

*void requestChecks(HTTPHeader *header, NaughtyFilter *checkme, String *urld, std::string *clientip, std::string *clientuser, int filtergroup, bool *ispostblock)* เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า header อยู่ใน blanket block, banned IP list, banned user list, banned URL list, banned regular expression URL list หรือไม่ และตรวจว่ามีการห้าม upload หรือขนาดของไฟล์ที่ upload เกินกว่าที่กำหนดหรือไม่

bool isIPHostnameStrip(String url) เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า *url* เป็นของเซิร์ฟเวอร์ที่โปรแกรมกำลังทำงานอยู่หรือไม่

*int determineGroup(std::string *user)* เป็นฟังก์ชั่นที่ใช้บอกว่า *user* อยู่ใน filter group ใด

*bool denyAccess (Socket *peerconn, Socket *proxysock, HTTPHeader *header, HTTPHeader *docheader, String *url, NaughtyFilter *checkme, std::string *clientuser, std::string *clientip, int filtergroup, bool ispostblock)* เป็นฟังก์ชั่นที่ใช้แสดงข้อความไม่อนุญาตให้ใช้งานหน้าเว็บกลับมาให้ client หรือกรณีที่กำหนดให้มี custom banned image ก็ให้แทนไฟล์รูปภาพด้วย custom banned image

String hashedURL(String *url, int filtergroup, std::string *clientip) เป็นฟังก์ชั่นที่ใช้สร้าง hashed URL จาก *url* ด้วย algorithm MD5

String hashedCookie(String *url, int filtergroup, std::string *clientip,. int bypasstimestamp) เป็นฟังก์ชั่นที่ใช้สร้าง hashed cookie จาก *url* ด้วย algorithm MD5

2.3.11 DataBuffer

เป็น class สำหรับเก็บและจัดการกับข้อมูลที่รับมาจากพรอกซี่เซิร์ฟเวอร์ มีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ตัวแปรและฟังก์ชั่นแบบ public

char data* เป็นตัวแปรที่ใช้เก็บข้อมูลที่ได้รับมาจากพรอกซี่เซิร์ฟเวอร์

int buffer_length เป็นตัวแปรที่ใช้เก็บขนาดของ *data*

char compresseddata* เป็นตัวแปรที่ใช้เก็บข้อมูลที่ได้รับมาจากพรอกซี่เซิร์ฟเวอร์ที่ถูก compress อยู่

int compressed_buffer_length เป็นตัวแปรที่ใช้เก็บขนาดของ *compresseddata*

DataBuffer() เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า default

~DataBuffer() เป็น destructor ของ class นี้ ใช้ลบค่าต่างๆ จากหน่วยความจำเพื่อป้องกันปัญหา memory leak

void read(Socket *sock, int length) throw(exception) เป็นฟังก์ชั่นที่ใช้อ่านข้อมูลจาก *sock* จำนวน *length* ไบต์และเก็บค่าไว้ใน *data*

int length() เป็นฟังก์ชั่นที่ใช้ return ค่า buffer_length

void copytomemory(char location)* เป็นฟังก์ชั่นที่ใช้ก๊อปปี้ค่าใน data ไปใส่ใน *location*

*bool in(Socket *sock)* เป็นฟังก์ชั่นที่ใช้อ่านข้อมูลจาก *sock* และเก็บค่าไว้ใน *data*

*void out(Socket *sock) throw(exception)* เป็นฟังก์ชั่นที่ใช้เขียนข้อมูลใน data ออกไปยัง *sock* และ throw exception เมื่อเกิดข้อผิดพลาด

void setTimeout(int t) เป็นฟังก์ชั่นที่ใช้กำหนดค่า *t* ให้กับตัวแปร *timeout*

void setDecompress(String d) เป็นฟังก์ชั่นที่ใช้กำหนดค่า *d* ให้กับตัวแปร *decompress*

bool contentRegExp(int filtergroup) เป็นฟังก์ชั่นที่ใช้แทนค่าข้อมูลใน data ด้วยข้อมูลใน content regular expression list ของ *filtergroup*

void swapbacktocompressed() เป็นฟังก์ชั่นที่ใช้กำหนดค่า *compresseddata* ให้กับ *data*

ตัวแปรและฟังก์ชั่นแบบ private

int timeout เป็นตัวแปรที่ใช้กำหนดค่า timeout ในการอ่านเขียน socket

String decompress เป็นตัวแปรที่ใช้บอกว่าข้อมูลถูกเข้ารหัสโดยวิธีใด (gzip หรือ deflate)

void zlibinflate(bool header) เป็นฟังก์ชั่นที่ใช้ uncompress ข้อมูลใน *data* โดยถ้า *header* เป็น true จะเป็น uncompress ด้วย gzip ถ้าเป็น false จะเป็นการ uncompress ด้วย zlib

2.3.12 DynamicURLList

เป็น class สำหรับเก็บและจัดการกับ URL list มีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ฟังก์ชั่นแบบ public

DynamicURLList() เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า default

~DynamicURLList() เป็น destructor ของ class นี้ ใช้ลบค่าต่างๆ จากหน่วยความจำเพื่อป้องกันปัญหา memory leak

bool setListSize(unsigned int s, unsigned int t) เป็นฟังก์ชั่นที่ใช้กำหนดค่าให้กับ *index, urlreftime, urls, size, timeout, agepos* โดย *s* กำหนดให้กับ *size* และ *t* กำหนดให้กับ *timeout*

void flush() เป็นฟังก์ชั่นที่ใช้กำหนดค่าให้ *urlreftime* ทุกค่าเป็น 0

bool inURLList(const char url)* เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า *url* อยู่ใน URL list cache หรือไม่

void addEntry(const char url)* เป็นฟังก์ชั่นที่ใช้เพิ่มค่า *url* เข้าไปใน URL list cache

ตัวแปรและฟังก์ชั่นแบบ private

unsigned int index* เป็นตัวแปรที่ใช้เก็บค่าว่าที่แต่ละตำแหน่งมี item อะไรอยู่

unsigned long int urlreftime* เป็นตัวแปรที่ใช้เก็บค่าว่าที่แต่ละตำแหน่งถูกเขียนเมื่อเวลาเท่าไร

char urls* เป็นตัวแปรที่ใช้เก็บค่า URL

int size เป็นตัวแปรที่ใช้เก็บขนาดของ URL list cache

int agepos เป็นตัวแปรที่ใช้เก็บตำแหน่งใน *index* ว่าตำแหน่งไหนมีอยู่มานานที่สุด

unsigned int timeout เป็นตัวแปรที่ใช้กำหนดอายุของ item ใน URL list cache ว่าจะมีอายุกี่วินาที

int items เป็นตัวแปรที่ใช้เก็บจำนวน item ที่อยู่ใน URL list cache

int search(int a, int s, const char url)* เป็นฟังก์ชั่นที่ใช้ค้นหาว่า *url* อยู่ใน URL list cache หรือไม่โดยใช้ binary search

int compare(const char a, const char* b)* เป็นฟังก์ชั่นที่ใช้เปรียบเทียบค่าของ *a* และ *b* ว่าเท่ากันหรือไม่ return ค่า 0 ถ้าเท่ากัน 1 ถ้า $a > b$ และ -1 ถ้า $a < b$

int posInList(const char url)* เป็นฟังก์ชั่นที่ใช้หาตำแหน่งของ url ใน URL list cache

2.3.13 String

เป็น class สำหรับเก็บและจัดการกับ URL list มีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ฟังก์ชั่นแบบ public

String() เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า default

~String() เป็น destructor ของ class นี้ ใช้ลบค่าต่างๆ จากหน่วยความจำเพื่อป้องกันปัญหา memory leak

String(const char bs[]) เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า *bs* ซึ่งเป็น C string (character array ที่ลงท้ายด้วย '\0')

String(const String& s) เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า *s*

String(const int num) เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า *num* ซึ่งเป็น integer

String(const long num) เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า *num* ซึ่งเป็น long integer

String(const char bs[], int len) เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า *bs* ซึ่งมีความยาว *len*

String(const char bs[], int start, int len) เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า *bs* โดยเริ่มจากตำแหน่ง *start* เป็นจำนวน *len* ตัวอักษร

friend ostream & operator << (ostream & out, const String& s) เป็นฟังก์ชั่นที่ใช้ overload operator << เพื่อเขียนค่า *s* ออกไปยัง *out*

friend String operator+ (const String & lhs, const String & s) เป็นฟังก์ชั่นที่ใช้ overload operator + เพื่อเชื่อม *lhs* และ *s* เข้าด้วยกัน

String& operator = (const String & s) เป็นฟังก์ชั่นที่ใช้ overload operator = เพื่อกำหนดค่าจาก *s*

bool operator != (const String & s) เป็นฟังก์ชั่นที่ใช้ overload operator != เพื่อเปรียบเทียบว่าไม่เท่ากัน

bool operator == (const String & s) เป็นฟังก์ชั่นที่ใช้ overload operator == เพื่อเปรียบเทียบว่าเท่ากัน

String& operator += (const String & s) เป็นฟังก์ชั่นที่ใช้ overload operator + เพื่อเชื่อม *s* เข้ามา

String operator+ (const String & s) เป็นฟังก์ชั่นที่ใช้ overload operator + เพื่อเชื่อม *s* เข้ามา

char operator [] (int i) const เป็นฟังก์ชั่นที่ใช้ overload operator [] เพื่อ return ค่าตัวอักษรที่ตำแหน่ง *i*

String after(const char bs)* เป็นฟังก์ชั่นที่ return String ตั้งแต่ *bs* ไปจนหมด

String before(const char bs)* เป็นฟังก์ชั่นที่ return String จากเริ่มต้นไปถึงก่อนหน้า *bs*

bool startsWith(const String s) เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า String เริ่มต้นด้วย *s* หรือไม่

bool endsWith(const String s) เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า String ลงท้ายด้วย *s* หรือไม่

String subString(int start, int l) เป็นฟังก์ชั่นที่ return subString จากตำแหน่ง *start* ไปเป็นจำนวน *l* ตัวอักษร

int toInteger() เป็นฟังก์ชั่นที่ใช้เปลี่ยนค่า String ให้เป็นตัวเลข integer

long int toLong() เป็นฟังก์ชั่นที่ใช้เปลี่ยนค่า String ให้เป็นตัวเลข long integer

int length() เป็นฟังก์ชั่นที่ return ค่าความยาวของ String

char toCharArray()* เป็นฟังก์ชั่นที่แปลง String ให้เป็น character array ที่ลงท้ายด้วย '\0'

*int indexOf(const char *s)* เป็นฟังก์ชั่นที่ return ตำแหน่งของ *s* ใน String

void chop() เป็นฟังก์ชั่นที่ใช้ลบตัวอักษรหนึ่งตัวออกจากท้าย String

void lop() เป็นฟังก์ชั่นที่ใช้ลบตัวอักษรหนึ่งตัวออกจากหน้า String

*bool contains(const char *s)* เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่ามี *s* อยู่ใน String หรือไม่

void toLower() เป็นฟังก์ชั่นที่ใช้เปลี่ยนตัวอักษรทั้งหมดให้เป็นตัวเล็ก

void toUpper() เป็นฟังก์ชั่นที่ใช้เปลี่ยนตัวอักษรทั้งหมดให้เป็นตัวใหญ่

void removeWhiteSpace() เป็นฟังก์ชั่นที่ใช้ลบ white space ออกจากหน้าและท้าย String

unsigned char charAt(int index) เป็นฟังก์ชั่นที่ return ค่าตัวอักษรที่ตำแหน่ง *index*

void removePTP() เป็นฟังก์ชั่นที่ใช้ลบ "http://", "ftp://", "https://" ออกจากหน้า String

int limitLength(unsigned int l) เป็นฟังก์ชั่นที่ใช้กำหนดค่า String ให้มีความยาว *l* ตัวอักษร

void removeMultiChar(unsigned char c) เป็นฟังก์ชั่นที่ใช้ลบตัวอักษร *c* ทั้งหมดออกจาก String

void hexDecode() เป็นฟังก์ชั่นที่ใช้ถอดรหัส hexadecimal string (แต่ละตัวอักษรขึ้นต้นด้วย % และตามด้วยรหัส ASCII ฐาน 16 ของตัวอักษร) ให้เป็น string ปกติ

String md5() เป็นฟังก์ชันที่ใช้คำนวณค่า MD5 ของ String

*String md5(const char *salt)* เป็นฟังก์ชันที่ใช้คำนวณค่า MD5 ของ String ต่อท้ายด้วย *salt*

*void replace(const char *what, const char *with)* เป็นฟังก์ชันที่ใช้แทนค่า *what* ด้วย *with*

void realPath() เป็นฟังก์ชันที่ใช้เปลี่ยนค่า String ให้เป็น path ที่ถูกต้อง

ตัวแปรแบบ private

char data* เป็นตัวแปรที่ใช้เก็บ String

int sl เป็นตัวแปรที่ใช้เก็บความยาวของ String

2.3.14 HTTPHeader

เป็น class สำหรับเก็บและจัดการกับ HTTP header มีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ตัวแปรและฟังก์ชั่นแบบ public

std::deque<String> header เป็นตัวแปรที่ใช้เก็บ HTTP header แต่ละบรรทัด

DataBuffer postdata เป็นตัวแปรที่ใช้อ่านเขียน HTTP header จาก socket

unsigned int port เป็นตัวแปรที่ใช้เก็บค่า port ที่อ่านได้จาก HTTP header

*void in(Socket *sock)* เป็นฟังก์ชั่นที่ใช้อ่านค่า HTTP header จาก *sock* แล้วเก็บค่าไว้ใน *header*

*void out(Socket *sock) throw(exception)* เป็นฟังก์ชั่นที่ใช้เขียนค่า *l* และ *postdata* ออกไปยัง *sock*

int contentlength() เป็นฟังก์ชั่นที่ return ค่าของ “Content-Length:” ที่กำหนดอยู่ใน *header*

bool iscontenttype(String t) เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า MIME type เท่ากับ *t* หรือไม่

bool malformedURL(String url) เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า *url* ถูกต้องหรือไม่

String url() เป็นฟังก์ชั่นที่ return ค่า url ของ *header* โดยดูจาก GET หรือ CONNECT method

String requesttype() เป็นฟังก์ชั่นที่ return ค่าชนิดของ method ของ *header* (GET, HEAD, PUT, etc)

String getcontenttype() เป็นฟังก์ชั่นที่ return ค่า MIME type ของข้อมูล

String disposition() เป็นฟังก์ชั่นที่ return ค่าของ “Content-Disposition:” ที่กำหนดอยู่ใน *header*

std::string getauthuser() เป็นฟังก์ชั่นที่ return ค่า user ที่กำหนดอยู่ใน *header*

std::string getXForwardedForIP() เป็นฟังก์ชั่นที่ return ค่าของ “X-Forwarded-For:” ที่กำหนดอยู่ใน *header*

void setTimeout(int t) เป็นฟังก์ชั่นที่กำหนดค่า timeout ให้เท่ากับ *t*

void addXForwardedFor(std::string clientip) เป็นฟังก์ชั่นที่เพิ่มค่า “X-Forwarded-For: *clientip*” เข้าไปใน *header*

bool isCompressed() เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า *header* นี้ถูก compress อยู่หรือไม่

String contentEncoding() เป็นฟังก์ชั่นที่ return ค่าของ “Content-Encoding:” ที่กำหนดอยู่ใน *header*

void removeEncoding(int newlen) เป็นฟังก์ชั่นที่ใช้ลบค่า “Content-Encoding:” ออกจาก *header* และกำหนดค่า *newlen* ให้กับ “Content-Length:”

bool isPostUpload() เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า POST method หรือไม่

bool isRedirection() เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่าเป็น redirection HTTP header หรือไม่

String decode(String s) เป็นฟังก์ชั่นที่ใช้ decode *s* ที่เข้ารหัสมาเป็น hexadecimal string

void setContentLength(int newlen) เป็นฟังก์ชั่นที่ใช้กำหนดค่าของ “Content-Length: *newlen*” ให้กับ *header*

bool authRequired() เป็นฟังก์ชั่นที่ใช้ตรวจ *header* ว่าต้องมีการ authentication หรือไม่

*int isBypassURL(String *url, const char *magic, const char *clientip)* เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่าใน *url* มี “GBYPASS=” อยู่หรือไม่

*bool isBypassCookie(String *url, const char *magic, const char *clientip)* เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่าใน *url* มี “GBYPASS” อยู่หรือไม่

void chopBypass(String url) เป็นฟังก์ชั่นที่ใช้ตัด “?GBYPASS=” หรือ “&GBYPASS=” ออกจาก *url*

*void setCookie(const char *cookie, const char *value)* เป็นฟังก์ชั่นที่ใช้เพิ่มค่า cookie เข้าไปใน *header*

ตัวแปรและฟังก์ชั่นแบบ private

void checkheader() เป็นฟังก์ชั่นที่ใช้ตรวจค่าใน *header* ว่าถูกต้องหรือไม่และแก้ไขให้ถูกต้อง

int timeout เป็นตัวแปรที่ใช้เก็บค่า timeout ในการอ่านเขียน socket

String getauth() เป็นฟังก์ชั่นที่ return ค่าของ "Proxy-Authentication: Basic "

String hexToChar(String n) เป็นฟังก์ชั่นที่ return ค่าตัวอักษรของ *n* ที่อยู่ในรูป hexadecimal string

int decode1b64(char c) เป็นฟังก์ชั่นที่ถูกเรียกใช้โดย decodeb64 เพื่อถอดรหัส base 64

std::string decodeb64(String line) เป็นฟังก์ชั่นที่ใช้ถอดรหัส base 64 ของ *line*

String modifyEncodings(String e) เป็นฟังก์ชั่นที่ใช้เพื่อตรวจดูว่า *e* มีการเข้ารหัสแบบใด

*String getCookie(const char *cookie)* เป็นฟังก์ชั่น return ค่า cookie จาก *cookie*

2.3.15 HTMLTemplate

เป็น class สำหรับเก็บและจัดการกับ HTML template โดยจะแทนค่าข้อมูลที่ต้องการแสดงผลเข้าไปใน template มีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ฟังก์ชั่นแบบ public

void reset() เป็นฟังก์ชั่นที่ใช้ลบข้อมูลทั้งหมดในตัวแปร *html*

bool readTemplateFile(const char filename)* เป็นฟังก์ชั่นที่ใช้อ่าน template file *filename* ขึ้นมาทีละบรรทัด และทำการค้นหาว่าตรงตำแหน่งไหนในบรรทัดบ้างที่มีคำว่า -URL- หรือ -REASONGIVEN- หรือ -REASONLOGGED- หรือ -USER- หรือ -IP- หรือ -FILTERGROUP- หรือ -BYPASS- ถ้ามีก็จะแยกคำเหล่านี้ออกจากอื่นก่อนที่จะใส่ทั้งหมดเข้าไปใน *html* ตามลำดับ

void display(Socket s, String url, String reason, String logreason, String user, String ip, String filtergroup, String hashed)* เป็นฟังก์ชั่นที่ใช้แทนค่าภายในตัวแปร *html* โดยแทนคำว่า -URL- ด้วย *url*, -REASONGIVEN- ด้วย *reason*, -REASONLOGGED- ด้วย *logreason*, -USER- ด้วย *user*, -IP- ด้วย *ip*, -FILTERGROUP- ด้วย *filtergroup* และ -BYPASS- ด้วย *url* + "?" + *hashed* หรือ *url* + "&" + *hashed* และเขียนค่าที่แก้ไขแล้วลงใน *s*

ตัวแปรและฟังก์ชั่นแบบ private

std::deque<String> html เป็นตัวแปรที่ใช้เก็บข้อมูลที่อ่านขึ้นมาจาก html template

void push(String s) เป็นฟังก์ชั่นที่ใช้เพิ่มข้อมูล *s* เข้าไปใน *html*

2.3.16 Ident

เป็น class สำหรับเก็บและจัดการกับตรวจสอบ username ด้วยวิธีการต่างตามที่กำหนดใน configuration file มีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ฟังก์ชั่นแบบ public

Ident() เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า default

std::string getUsername(HTTPHeader h, string* s, int port)* เป็นฟังก์ชั่นที่ return ค่า username จาก *h* ซึ่งจะเป็นตัวอักษรตัวเล็กโดยการเรียกใช้ *getUsernameProxyAuth()* หากตัวแปร *o.uim_proxyauth* เป็น 1 หรือเรียกใช้ *getUsernameNTLM()* หากตัวแปร *o.uim_ntlm* เป็น 1 หรือเรียกใช้ *getUsernameIdent* หากตัวแปร *o.uim_ident* เป็น 1 แต่ถ้าไม่สามารถหาค่า username ได้ก็จะ return ค่าเป็น "-"

ตัวแปรและฟังก์ชั่นแบบ private

bool getUsernameProxyAuth(HTTPHeader h)* เป็นฟังก์ชันที่ใช้หา username จาก h โดยจะกำหนดค่า *username* ตามที่อยู่ใน *h* และ return ค่า true แต่ถ้าไม่มี username อยู่ใน *h* จะ return ค่า false

bool getUsernameNTLM(HTTPHeader h)* เป็นฟังก์ชันที่ใช้หา username ด้วยวิธี NTLM (แต่ยังไม่ได้ implement จึง return ค่าเป็น false เสมอ)

bool getUsernameIdent(string s, int port, int serverport, int timeout)* เป็นฟังก์ชันที่ใช้หา username จาก identd server *s* โดยใช้ *port* และ *serverport* ที่กำหนด และจะรอเป็นเวลา *timeout* วินาที

std::string username เป็นตัวแปรที่ใช้เก็บ username ที่ได้จากการหาด้วยวิธีต่างๆ

2.3.17 ImageContainer

เป็น class สำหรับเก็บและจัดการกับรูปภาพที่จะใช้แสดงผลให้ client แทนถ้ามีการกำหนดไว้ใน configuration file มีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ฟังก์ชั่นแบบ public

ImageContainer() เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า default

~ImageContainer() เป็น destructor ของ class นี้ ใช้ลบค่าต่างๆ จากหน่วยความจำเพื่อป้องกันปัญหา memory leak

void reset() เป็นฟังก์ชั่นที่ใช้ลบข้อมูลออกจากหน่วยความจำ

bool read(const char filename)* เป็นฟังก์ชันที่ใช้อ่านไฟล์รูปภาพ *filename* ขึ้นมาเก็บไว้ใน *image* เก็บความยาวของไฟล์ไว้ใน *imagelength* และเก็บ MIME type ไว้ใน *mimetype*

void display(Socket s)* เป็นฟังก์ชั่นที่ใช้เขียนข้อมูลใน *image* ออกไปยัง Socket *s*

ตัวแปรแบบ private

long int imagelength เป็นตัวแปรที่ใช้เก็บความยาวของรูปภาพ

String mimetype เป็นตัวแปรที่ใช้เก็บ MIME type ของรูปภาพ

*char *image* เป็นตัวแปรที่ใช้เก็บรูปภาพ

2.3.18 FDTunnel

เป็น class สำหรับจัดการในการส่งผ่านข้อมูลจากพรอกซี่เซิร์ฟเวอร์ไปยัง client และ client ไปยังพรอกซี่เซิร์ฟเวอร์โดยตรงโดยไม่มีการเปลี่ยนแปลงข้อมูลภายใน ซึ่งมีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ตัวแปรและฟังก์ชั่นแบบ public

FDTunnel() เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า default

void tunnel(int fdfrom, int fdto) เป็นฟังก์ชั่นที่ใช้ในการส่งผ่านข้อมูลจาก descriptor *fdfrom* ไปยัง descriptor *fdto*

int throughput เป็นตัวแปรที่ใช้เก็บจำนวนข้อมูลทั้งหมดที่ส่งผ่านจาก fdfrom ไปยัง fdto

ตัวแปรและฟังก์ชั่นแบบ private

int read(int fd, char buff, int len)* เป็นฟังก์ชั่นที่ใช้อ่านข้อมูลจาก descriptor *fd* เข้ามาใส่ใน *buff* ที่มีขนาด *len*

bool write(int fd, char* buff, int len) เป็นฟังก์ชั่นที่ใช้เขียนข้อมูลจาก *buff* ขนาด *len* ออกไปยัง descriptor *fd*

*int selectEINTR(int numfds, fd_set * readfds, fd_set * writefds, fd_set * exceptfds, struct timeval * timeout)* เป็นฟังก์ชั่นที่ใช้เรียกฟังก์ชั่น select โดยจะมีการเรียกซ้ำถ้าได้รับ signal

2.3.19 LanguageContainer

เป็น class สำหรับจัดการในการแปลรหัสข้อผิดพลาดต่างๆ ให้เป็นภาษาที่กำหนดไว้ใน configuration file ซึ่งมีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ฟังก์ชั่นแบบ public

void reset() เป็นฟังก์ชั่นที่ใช้ลบค่าที่อยู่ใน *keys* และ *values*

bool readLanguageList(const char filename)* เป็นฟังก์ชั่นที่ใช้อ่าน message จาก filename และแบ่งค่าในแต่ละบรรรทัดออกมาเป็น *key* และ *values* เพื่อมาใส่ไว้ในตัวแปรทั้งสอง

const char* getTranslation(const unsigned int index) เป็นฟังก์ชันที่ return ค่าเป็น *value* ที่มี *key* ตรงตาม index ที่กำหนด

ตัวแปรแบบ private

std::deque<unsigned int> keys เป็นตัวแปรที่ใช้เก็บค่า *key* ของแต่ละ message

std::deque<String> values เป็นตัวแปรที่ใช้เก็บค่า *value* ของแต่ละ message

2.3.20 RegExp

เป็น class สำหรับจัดการในการใช้งาน regular regular expression ซึ่งมีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ฟังก์ชั่นแบบ public

RegExp() เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า default

~RegExp() เป็น destructor ของ class นี้ ใช้ลบค่าต่างๆ จากหน่วยความจำเพื่อป้องกันปัญหา memory leak

RegExp(const RegExp& r) เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตาม *r*

bool comp(const char exp)* เป็นฟังก์ชั่นที่ใช้ compile regular expression *exp* และเก็บค่าไว้ในตัวแปร *reg*

bool match(const char text)* เป็นฟังก์ชั่นที่ใช้เพื่อหาว่าพบ *text* ใน *reg* หรือไม่

int numberOfMatches() เป็นฟังก์ชั่นที่ใช้หาว่ามีการค้นหา แล้วพบทั้งหมดกี่ครั้ง

bool matched() เป็นฟังก์ชั่นที่ใช้ตอบว่าเคยมีการค้นหา แล้วพบหรือยัง

std::string result(int i) เป็นฟังก์ชั่นที่ใช้ตอบผลของการการค้นหาแล้วพบครั้งที่ *i*

unsigned int offset(int i) เป็นฟังก์ชั่นที่ใช้บอกตำแหน่ง offset ของการค้นหาแล้วพบครั้งที่ *i*

unsigned int length(int i) เป็นฟังก์ชั่นที่ใช้บอกขนาดของผลการค้นหาแล้วพบครั้งที่ *i*

char search(char* file, char* fileend, char* phrase, char* phraseend)* เป็นฟังก์ชั่นที่ใช้ค้นหาว่ามีวลีที่ชี้โดย *phrase* และลงท้ายด้วยที่ตำแหน่ง *phraseend* อยู่ใน *file* ที่ลงท้ายด้วยตำแหน่ง *fileend* หรือไม่ โดยการค้นหาใช้ Boyer Moore quick search algorithm

ตัวแปรแบบ private

std::deque<std::string> results เป็นตัวแปรที่ใช้เก็บผลของการค้นหา

std::deque<unsigned int> offsets เป็นตัวแปรที่ใช้เก็บตำแหน่งที่ค้นหาพบ

std::deque<unsigned int> lengths เป็นตัวแปรที่ใช้เก็บขนาดของผลการค้นหาแต่ละครั้ง

bool imatched เป็นตัวแปรที่ใช้บอกว่าเคยมีการค้นหาพบแล้วหรือยัง

regex_t reg เป็นตัวแปรที่ใช้เก็บ regular expression ที่ compile แล้ว

bool wascompiled เป็นตัวแปรที่ใช้บอกว่ามีการ compile regular expression แล้วหรือยัง

std::string searchstring เป็นตัวแปรที่ใช้เก็บ string ที่จะใช้เพื่อให้มาค้นหา

2.3.21 MD5

เป็นโมดูลสำหรับจัดการในการคำนวณ MD5 message digest ตามที่กำหนดใน RFC1321

2.3.22 NaughtyFilter

เป็น class สำหรับจัดการในการทำ content filtering ซึ่งมีตัวแปรและฟังก์ชั่นต่างๆ ดังต่อไปนี้

ตัวแปรและฟังก์ชั่นแบบ public

NaughtyFilter() เป็น constructor ของ class นี้ ใช้เพื่อกำหนดค่าเริ่มต้นตามค่า default

void checkme(DataBuffer body)* เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่าข้อมูลใน *body* มีความเหมาะสมที่จะให้ client ใช้งานได้หรือไม่ ซึ่งมีรายละเอียดในการทำงานดังนี้

1) ทำการก๊อปปี้ข้อมูลใน body ไปเก็บไว้ในตัวแปร bodylc โดยจะเปลี่ยนให้เป็นตัวอักษรตัวเล็กทั้งหมด ยกเว้นถ้าใน configuration file กำหนดให้ไม่ต้องเปลี่ยน และแทนที่ white space ด้วย space
2) ถ้าใน configuration file กำหนดให้มีการถอดรหัสข้อมูลที่เป็น hexadecimal character ก็ทำการถอดรหัส
3) ตรวจสอบข้อมูลด้วยการเรียกฟังก์ชั่น *checkPICS()* ถ้าไม่ผ่านก็ให้ออกจากการทำงาน
4) ตรวจสอบข้อมูลด้วยการเรียกฟังก์ชั่น *checkphrase()* ถ้าไม่ผ่านก็ให้ออกจากการทำงาน
5) ทำการตัด html tags ออกจากข้อมูลแล้วตรวจสอบอีกครั้งด้วยการเรียกฟังก์ชั่น *checkphrase()* แล้วจบการทำงาน

bool isItNaughty เป็นตัวแปรที่ใช้บอกว่าข้อมูลที่กำลังตรวจสอบมีเนื้อหาที่ไม่เหมาะสมหรือไม่

bool isException เป็นตัวแปรที่ใช้บอกว่าข้อมูลที่กำลังตรวจสอบอยู่ในข้อยกเว้นหรือไม่

int filtergroup เป็นตัวแปรที่ใช้บอกว่าข้อมูลที่กำลังตรวจสอบนี้จะตรวจสอบโดยใช้ filtergroup ใด

std::string whatIsNaughty เป็นตัวแปรที่ใช้เก็บคำอธิบายว่าทำไมถึงไม่อนุญาตให้ใช้งานข้อมูลที่กำลังตรวจสอบนี้ เพื่อใช้แสดงผลให้ client

std::string whatIsNaughtyLog เป็นตัวแปรที่ใช้เก็บคำอธิบายว่าทำไมถึงไม่อนุญาตให้ใช้งานข้อมูลที่กำลังตรวจสอบนี้ เพื่อใช้เขียนลง log file

ตัวแปรและฟังก์ชั่นแบบ private

void checkphrase(char file, int l)* เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่าข้อมูลใน *file* ที่มีความยาว *l* นั้นมีเนื้อหาที่ไม่เหมาะสมหรือไม่ โดยจะมีการกำหนดค่าให้กับตัวแปร *isItNaughty, isException, whatIsNaughty* และ *whatIsNaughtyLog* ตามที่เหมาะสม โดยมีรายละเอียดการทำงานดังนี้

1) ค้นหาข้อมูลใน *file* ว่ามีวลีที่ไม่เหมาะสมหรือไม่ และเก็บผลที่ได้ไว้ในตัวแปร *found*
2) ตรวจสอบใน combination list ว่ามีข้อยกเว้น (exception) หรือต้องคำนวณน้ำหนักและกำหนดค่าให้กับตัวแปรที่เกี่ยวข้อง
3) ตรวจดูข้อมูลใน *found* เพื่อคำนวณน้ำหนักตามวิธีการที่กำหนดใน configuration file
4) เมื่อพบวลีที่ไม่เหมาะสมก็กำหนดค่าให้กับ whatIsNaughty และ whatIsNaughtyLog ให้ถูกต้องแล้วออกจากการทำงาน

void checkPICS(char file, int l)* เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่าข้อมูลใน *file* ที่มีความยาว *l* นั้นมีการกำหนด rate ด้วย PICS หรือไม่ถ้ามีก็จะเรียกใช้ฟังก์ชั่น *checkPICSrating()* เพื่อตรวจสอบต่อไป

void checkPICSrating(std::string label) เป็นฟังก์ชั่นที่ใช้ตรวจสอบว่า PICS *label* เป็นประเภทใดและเรียกใช้ฟังชั่นตรวจสอบที่เหมาะสม

void checkPICSratingSafeSurf(String r) เป็นฟังก์ชั่นที่ใช้ตรวจสอบ rating *r* ว่าอยู่ใน rate ที่กำหนดหรือไม่

void checkPICSratingevaluWEB(String r) เป็นฟังก์ชั่นที่ใช้ตรวจสอบ rating *r* ว่าอยู่ใน rate ที่กำหนดหรือไม่

void checkPICSratingCyberNOT(String r) เป็นฟังก์ชั่นที่ใช้ตรวจสอบ rating *r* ว่าอยู่ใน rate ที่กำหนดหรือไม่

void checkPICSratingRSAC(String r) เป็นฟังก์ชั่นที่ใช้ตรวจสอบ rating *r* ว่าอยู่ใน rate ที่กำหนดหรือไม่

void checkPICSratingICRA(String r) เป็นฟังก์ชั่นที่ใช้ตรวจสอบ rating *r* ว่าอยู่ใน rate ที่กำหนดหรือไม่

void checkPICSratingWeburbia(String r) เป็นฟังก์ชั่นที่ใช้ตรวจสอบ rating *r* ว่าอยู่ใน rate ที่กำหนดหรือไม่

void checkPICSratingVancouver(String r) เป็นฟังก์ชั่นที่ใช้ตรวจสอบ rating *r* ว่าอยู่ใน rate ที่กำหนดหรือไม่

void checkPICSagainstoption(String s, char l, int opt, std::string m)* เป็นฟังก์ชั่นที่ใช้ตรวจสอบ rating *s* ว่า label *l* มีค่ามากกว่า *opt* หรือไม่ ถ้ามากกว่าแสดงว่าเกิน rate ที่กำหนดใน configuration file โดย *m* จะเป็นชนิดของ PICS rating

บทที่ 3
การออกแบบโปรแกรมป้องกันการเข้าถึงเว็บไซต์ภาษาไทยที่ไม่เหมาะสม

3.1 เป้าหมายในการออกแบบโปรแกรม

เนื่องจากจุดหมายของโครงการพัฒนาระบบงานนี้ ก็เพื่อจะที่จะออกแบบระบบให้สามารถทำการตรวจสอบและป้องกันการเข้าถึงเว็บไซต์ภาษาไทยที่ไม่เหมาะสม แต่เนื่องจากเว็บไซต์ที่มีเนื้อหาไม่เหมาะสมนั้นมีทั้งที่เป็นภาษาไทยและภาษาต่างประเทศ ดังนั้นเพื่อให้โปรแกรมสามารถใช้งานได้อย่างกว้างขวาง ตรวจสอบเว็บไซต์ที่ไม่เหมาะสมได้อย่างทั่วถึงและนำไปประยุกต์ใช้งานได้จริง จึงได้มีการออกแบบโปรแกรมโดยการเพิ่มขยายความสามารถให้โปรแกรม DansGuardian ซึ่งเป็นโปรแกรมที่ทำหน้าที่ตรวจสอบเนื้อหาของหน้าเว็บ ที่มีความสามารถตรวจสอบหน้าเว็บภาษาต่างประเทศได้ดีและมีการพัฒนาอย่างต่อเนื่อง การออกแบบจะเป็นการเพิ่มโมดูลตรวจสอบคำภาษาไทยเข้าไปในโปรแกรม DansGuardian และดัดแปลงโปรแกรมบางส่วนให้ทำงานกับภาษาไทยได้ถูกต้อง เนื่องจากการทำงานในหลายๆ ส่วนของ DansGuardian มีการออกแบบที่ทำให้ไม่สามารถตรวจสอบภาษาไทยได้

ความต้องการพื้นฐานในออกแบบการทำงานของโปรแกรม ที่จะได้รับการพัฒนาเพิ่มเติมและดัดแปลงขึ้น มีดังต่อไปนี้

- ทำการศึกษาการทำงานทั้งหมดของโปรแกรม DansGuardian เพื่อหาจุดที่จะสามารถเชื่อมต่อการเพิ่มเติมความสามารถใจการตรวจหน้าเว็บภาษาไทยได้
- พัฒนาโมดูลเพิ่มเติมเพื่อเชื่อมต่อเข้ากับโปรแกรม DansGuardian ให้สามารถรองรับการตรวจหน้าเว็บภาษาไทย
- ทำการจัดสร้างบัญชีดำของวลีภาษาไทยที่ไม่เหมาะสม เพื่อให้โมดูลที่พัฒนาเพิ่มเติมเข้ามาสามารถตรวจหน้าเว็บภาษาไทยที่มีวลีเหล่านั้นได้
- ทำการจัดสร้าง HTML template และข้อความเตือนต่างๆ ให้เป็นภาษาไทย เพื่อที่ผู้ใช้งานคนไทยจะได้เข้าใจได้ง่ายยิ่งขึ้น
- ระบบตรวจจับจะต้องสามารถตรวจหาวลีที่ไม่เหมาะสม ที่ผู้จัดทำหน้าเว็บพยายามทำให้หลุดรอดจากการตรวจจับโดยการเพิ่มช่องว่างหรือตัวอักษรอื่นๆ เข้าไปในระหว่างคำได้

3.2 การออกแบบการทำงานของโปรแกรม

ก่อนที่จะได้ทำการศึกษาการทำงานของโปรแกรม DansGuardian อย่างละเอียดนั้น ผู้จัดทำโครงการได้วางแผนที่จะทำการตรวจสอบภาษาไทยในหน้าเว็บโดยการตัดแบ่งหน้าเว็บออกเป็นคำๆ แล้วนำไปค้นหาในตารางที่เก็บบัญชีคำรายการของคำที่ไม่เหมาะสม แต่จากการศึกษาวิธีการตัดคำภาษาไทยต่างๆ ไม่ว่าจะเป็นการตัดคำด้วยพจนานุกรม หรือการตัดคำด้วยอัลกอริทึม พบว่าไม่สามารถทำงานได้ถูกต้องมากนักโดยเฉพาะอย่างยิ่งอัลกอริทึมในการตัดคำมักไม่รู้จักคำแสลง ทำให้ทำงานได้ไม่ดีเท่าที่ควร เมื่อได้ศึกษาการทำงานโดยละเอียดของโมดูลย่อยต่างๆ ทั้ง 22 โมดูลของ DansGuardian แล้วก็พบว่า มีการใช้วิธีการที่ดีกว่า คือแทนที่จะแบ่งหน้าเว็บออกเป็นคำๆ แล้วนำไปค้นหาในตารางบัญชีดำ เราจะใช้วลีหรือคำในบัญชีดำมาค้นหาในหน้าเว็บแทน โดยการค้นหานั้นจะใช้อัลกอริทึม Boyer Moore quick search ซึ่งสามารถค้นหาคำได้อย่างรวดเร็วและไม่จำเป็นต้องแบ่งหน้าเว็บออกเป็นคำๆ ด้วย จึงสามารถใช้อัลกอริทึมนี้ในการนำวลีที่ไม่เหมาะสมภาษาไทยมาค้นหาในหน้าเว็บได้ทันที

การทำงานของโปรแกรมส่วนที่เกี่ยวข้องกับการตรวจหาวลีที่ไม่เหมาะสมนั้น อยู่ในโมดูล NaughtyFilter ฟังก์ชั่น *checkme()* จากการศึกษาพบว่าสาเหตุที่โปรแกรมเดิมไม่สามารถตรวจสอบวลีที่เป็นภาษาไทยได้นั้น ก็เนื่องจากโปรแกรมจะทำการเปลี่ยนตัวอักษรทุกตัวในหน้าเว็บให้เป็นตัวอักษรตัวเล็ก โดยรวมไปถึงตัวอักษรที่มีพื้นฐานมาจากภาษาลาตินที่มีเครื่องหมาย accent อยู่ด้านบนด้วย และตัวอักษรที่มี accent เหล่านี้เองที่มีรหัสเดียวกันกับภาษาไทย ดังแสดงในตารางที่ 3.1

จะเห็นได้ว่าเมื่อโปรแกรมเปลี่ยนตัวอักษรลาติน “À” จากตัวใหญ่ให้กลายเป็นตัวเล็ก “à” ก็จะทำให้หน้าเว็บที่มีตัวอักษรภาษาไทย “ภ” กลายเป็น “ำ” ไปด้วย จึงไม่สามารถตรวจจับวลีที่ไม่เหมาะสมภาษาไทยได้ แต่ DansGuardian ก็อนุญาตให้ไม่ต้องเปลี่ยนตัวอักษรให้เป็นตัวเล็กได้ แต่ถ้าจะไม่เปลี่ยนตัวอักษรตัวใหญ่ให้เป็นตัวเล็กแล้ว การตรวจจับวลีในภาษาต่างประเทศที่มีพื้นฐานมาจากภาษาลาตินก็จะทำได้ยากมาก ตัวอย่างเช่นถ้าต้องการตรวจคำว่า sex ให้ครบทุกกรณี จะต้องมีการตรวจคำเหล่านี้ทั้งหมด sex Sex sEx seX SEx SeX sEX SEX ดังนั้นในทางปฏิบัติจึงต้องมีการเปลี่ยนตัวอักษรทั้งหมดให้เป็นตัวเล็กเพื่อให้ตรวจสอบได้ครอบคลุมที่สุด

ตารางที่ 3.1 ตัวอย่างรหัสตัวอักษรภาษาไทยและลาติน

รหัสตัวอักษร	ตัวอักษรลาติน	ตัวอักษรไทย
192	À	ภ
193	Á	ม
194	Â	ย
195	Ã	ร
224	à	เ
225	á	แ
226	â	โ
227	ã	ใ

การออกแบบโปรแกรมให้สามารถตรวจสอบวลีภาษาไทยได้ จึงเริ่มจากการนำข้อมูลดิบที่ยังไม่ได้มีการเปลี่ยนให้เป็นตัวเล็ก มาทำการตัดตัวอักษรที่ไม่ใช่ภาษาไทยออกไปก่อน (โดยตัวอักษรไทยจะมีรหัสตั้งแต่ 161 ขึ้นไป) เพราะโปรแกรมของเราที่จะพัฒนาเพิ่มจะสนใจเพียงแค่ภาษาไทยเท่านั้น นอกจากนี้ส่วนที่ไม่ใช่ภาษาไทยก็จะถูกตรวจสอบโดยโปรแกรมหลักตามปกติอยู่แล้ว จึงไม่มีความจำเป็นที่จะต้องมาตรวจสอบซ้ำซ้อน และการที่ตัดตัวอักษรที่ไม่ใช่ภาษาไทยออกไป ยังทำให้สามารถตรวจพบวลีที่มีการเขียนโดยแทรกตัวอักษรอื่นเข้าไประหว่างกลางได้ด้วย ดังตัวอย่างในรูปที่ 3.1

โ-ค-ต-ร

รูปที่ 3.1 ตัวอย่างวลีที่มีการแทรกตัวอักษรอื่นเข้าไปด้วย

ซึ่งโดยปกติแล้วหากไม่มีการตัดตัวอักษรอื่นที่แทรกเข้าไป (ในที่นี้คือ "-") ก็จะไม่สามารถตรวจพบคำนี้ในการค้นหาได้

3.2.1 ไฟล์ที่ต้องแก้ไขใน DansGuardian

การออกแบบระบบให้สามารถตรวจสอบภาษาไทยได้ จะมีการแก้ไขไฟล์ที่เกี่ยวข้องของโปรแกรม DansGuardian ทั้งหมด 4 ไฟล์ด้วยกันดังต่อไปนี้คือ

1) configure เป็น shell script ที่ใช้ในการสร้าง makefile ให้เหมาะสมกับแต่ละระบบปฏิบัติการ การแก้ไขจะมีการเพิ่มคำสั่งในรูปที่ 3.2 และ 3.3 เข้าไปเพื่อให้การติดตั้งโปรแกรมถูกต้อง

```
echo "" >>dansguardian.conf
echo "" >>dansguardian.conf
echo "" >>dansguardian.conf
echo "# Added for Thai filtering." >>dansguardian.conf
echo "#" >>dansguardian.conf
echo "# Evenghough increasing CPU load but work properly with Thai." >>dansguardian.conf
echo "# on|off (defaults to off)" >>dansguardian.conf
echo "thaifiltering = off" >>dansguardian.conf
```

รูปที่ 3.2 คำสั่งในการเพิ่มพารามิเตอร์ thaifiltering เข้าไปใน dansguardian.conf

```
$extendedecho ".Include<$prefixdir$sysconfdir\c" >>weightedphraselist
echo "phraselists/pornography/weighted_thai> #extremely ALPHA#" >>weightedphraselist
```

รูปที่ 3.3 คำสั่งในการเพิ่มการตรวจสอบภาษาไทยเข้าไปในไฟล์ weighedphraselist

2) OptionContainer.cpp เป็นโปรแกรมที่ใช้ในการอ่านค่าจาก configuration file เข้ามาและกำหนดค่าให้กับตัวแปรเพื่อทำงาน ในส่วนนี้จะมีการเพิ่มคำสั่งในรูปที่ 3.4 เข้าไปเพื่อที่จะกำหนดค่าให้กับโปรแกรมว่าต้องตรวจสอบภาษาไทยหรือไม่

```
// To get Thai filtering flag
    If (findoptionS("thaifiltering") == "on") {
       enablethai = true;
    }else {
       enablethai = false;
    }
```

รูปที่ 3.4 คำสั่งในการกำหนดค่าให้กับตัวแปรเพื่อตรวจสอบภาษาไทย

3) OptionContainer.hpp เป็น Header file ของ OptionContainer.cpp สิ่งที่กำหนดเพิ่มในไฟล์นี้ก็คือกำหนดตัวแปร public enablethai เพิ่มเติมเข้ามาดังรูปที่ 3.5

```
bool enablethai; // enable/disable Thai filtering
```

รูปที่ 3.5 คำสั่งเพื่อเพิ่มตัวแปรสำหรับบอกว่าจะมีการตรวจสอบภาษาไทยหรือไม่

4) NaughtyFilter.cpp เป็นส่วนของโปรแกรมที่เราจะนำส่วนตรวจสอบภาษาไทยใส่เพิ่มเข้าไป โดยมีคำสั่งเพิ่มเติมที่ใส่เข้าไปในฟังก์ชั่น *checkme()* ดังรูปที่ 3.6 และต้องมีการ include header file ThaiProcessing.hpp เข้าไปในไฟล์นี้ด้วย

```
if (o.enablethai) {
    ThaiProcessing thaiobject; // to hold document for Thai processing.
    thaiobject.extractThai(rawbody, bodylen);
    if (thaiobject.getLength() > 1) { // word should have more than 1 character, right?
      checkphrase((char *)thaiobject.getData(), thaiobject.getLength());  // check raw
      if (isItNaughty || isException) {
        return;  // Well, there is no point in continuing is there?
      }
    }
}
```

รูปที่ 3.6 คำสั่งในการเพิ่มการตรวจสอบภาษาไทยเข้าไปในไฟล์ NaughtyFilter.cpp

3.2.2 ไฟล์ที่ต้องสร้างเพิ่มเติม

นอกจากนี้ยังมีการเพิ่มไฟล์ต่อไปนี้เข้าไปในระบบด้วยเพื่อให้สามารถตรวจสอบและแสดงผลเป็นภาษาไทยได้ถูกต้อง

1) language/thai/message เป็นไฟล์ที่เก็บรหัสแสดงข้อผิดพลาด และข้อความแสดงข้อผิดพลาด
2) language/thai/template.html เป็นไฟล์ที่เก็บ HTML template สำหรับใช้ในการแสดงผลบนเว็บบราวเซอร์ของผู้ใช้
3) phraselists/pornography/weighted_thai เป็นไฟล์ที่เก็บบัญชีคำของวลีภาษาไทย รวมทั้งน้ำหนักของแต่ละวลีด้วย นอกจากไฟล์นี้แล้วเรายังสามารถเพิ่มเติมไฟล์อื่นๆ เข้าไปได้อีกในไดเรกทอรี phraselists เพื่อให้แบ่งเป็นหมวดหมู่ สำหรับการตรวจสอบวลีที่ไม่เหมาะสมประเภทต่างๆ และก็ต้องเพิ่มคำสั่งเข้าไปในไฟล์ configure ด้วยเช่นกัน เพื่อให้มีการเรียกใช้ไฟล์ที่เราเพิ่มเข้ามา
4) ThaiProcessing.cpp เป็นโปรแกรมที่เราใช้ในการคัดเอาเฉพาะตัวอักษรภาษาไทยออกมาจากหน้าเว็บ ประกอบด้วย 3 ฟังก์ชั่นคือ
    a. *ThaiProcessing()* เป็น constructor ของ class นี้ ทำหน้าที่กำหนดค่าเริ่มต้นให้กับตัวแปรต่างๆ ตามค่า default

b. *~ThaiProcessing()* เป็น destructor ของ class นี้ ใช้ลบค่าต่างๆ จากหน่วยความจำเพื่อป้องกันปัญหา memory leak

c. extractThai(const char* sourcedata, int sourcelen) เป็นฟังก์ชั่นที่ใช้ในการคัดเอาเฉพาะตัวอักษรภาษาไทยออกมาจากหน้าเว็บ เพื่อนำไปเก็บไว้ในตัวแปร *thaidata* และความยาวเก็บไว้ในตัวแปร *thailen*

5) ThaiProcessing.hpp เป็น header file ของ ThaiProcessing.cpp ทำหน้าที่กำหนดตัวแปรและฟังก์ชั่นต่างๆ ที่ใช้ใน class ThaiProcessing.hpp

ตารางที่ 3.2 สรุปไฟล์ต่างๆ ที่ต้องแก้ไขและสร้างเพิ่มเติม

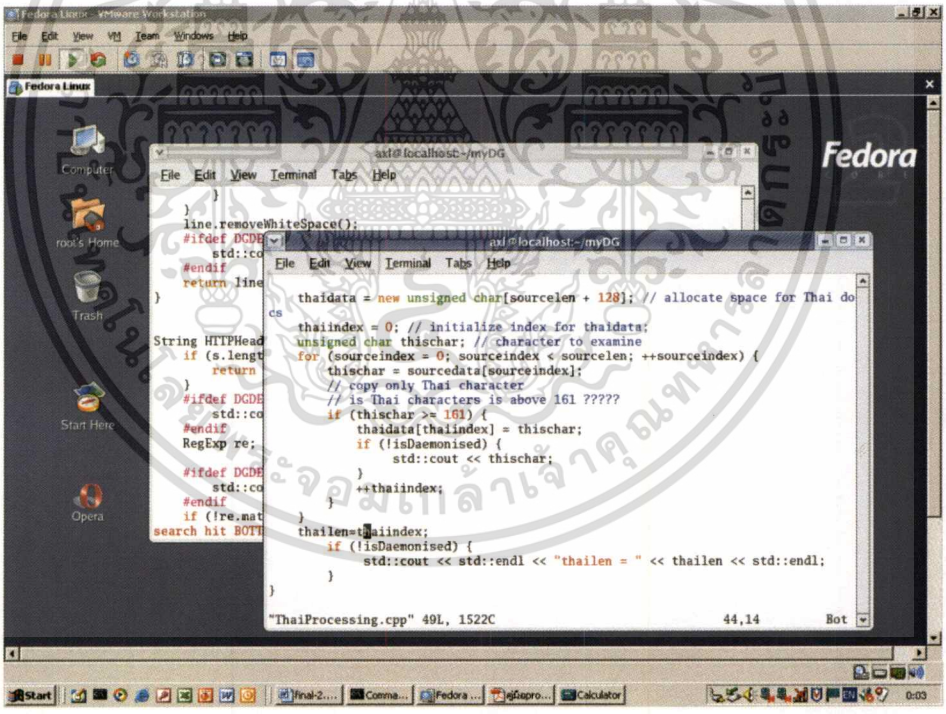
1. configure
2. OptionContainer.cpp
3. OptionContainer.hpp
4. NaughtyFilter.cpp
5. language/thai/message
6. language/thai/template.html
7. phraselists/pornography/weighted_thai
8. ThaiProcessing.cpp
9. ThaiProcessing.hpp

บทที่ 4
เครื่องมือต่างๆ ที่ใช้ในการพัฒนาโปรแกรม

4.1 เครื่องมือที่ใช้ในการพัฒนาโปรแกรม

1) โปรแกรม VMware Workstation เวอร์ชั่น 5.0.0

เป็นโปรแกรมที่ใช้สำหรับสร้างเครื่องคอมพิวเตอร์เสมือนขึ้นมาเพื่อใช้รันระบบปฏิบัติการ Linux Fedora Core 2 โดยทำงานภายใต้ MS Windows 2000 อีกทีหนึ่ง ทำให้ไม่จำเป็นต้องใช้คอมพิวเตอร์หลายเครื่องในการพัฒนาโปรแกรม



รูปที่ 4.1 หน้าจอของโปรแกรม VMwarae

2) โปรแกรม vim

เป็นโปรแกรม text editor ที่ใช้ในการเขียนโปรแกรมและทั้งหมด และใช้ในการเปลี่ยนแปลงแก้ไขไฟล์ต่างๆ ที่เกี่ยวข้อง

3) โปรแกรม GNU C++ compiler เวอร์ชั่น 3.3.3

เป็น compiler ภาษา C++ ที่ทำงานบนระบบปฏิบัติการ Linux ใช้ในการ compile โปรแกรมทั้งหมด

4) โปรแกรม GNU make เวอร์ชั่น 3.80

เป็นโปรแกรม make ที่ใช้ในการสั่งการ compile โปรแกรมทั้งหมด และยังใช้ในการติดตั้งโปรแกรมที่ compile เสร็จแล้วลงในเครื่องด้วย

5) โปรแกรม DansGuardian เวอร์ชั่น 2.8.0.6

เป็นโปรแกรมที่ทำหน้าที่ตรวจสอบข้อมูลในหน้าเว็บที่ผู้ใช้ต้องการเข้าถึง ว่ามีข้อความที่ไม่เหมาะสมอยู่หรือไม่ ถ้ามีก็จะป้องกันไม่ให้ผู้ใช้เข้าใช้งานและแจ้งคำเตือนกลับไป เป็นโปรแกรมที่ใช้เป็นพื้นฐานในการพัฒนาเพิ่มเติมเพื่อให้สามารถตรวจสอบหน้าเว็บที่เป็นภาษาไทยได้

4.2 ขั้นตอนในการพัฒนาโปรแกรม

เนื่องจากไม่ได้เขียนโปรแกรมด้วยภาษา C++ มาเป็นเวลานานแล้ว ดังนั้นก่อนที่จะเริ่มงานจริงๆ จึงต้องทำการศึกษาการเขียนโปรแกรมด้วยภาษา C++ ใหม่โดยได้ศึกษาจากหนังสือ Practical C++ Programming และเนื่องจากโปรแกรมที่จะพัฒนาต้องทำงานอยู่บนระบบปฏิบัติการ unix หรือ linux จึงได้ศึกษาเพิ่มเติมจากหนังสือ Advanced Programming in the UNIX Environment ด้วย ในตอนแรกตั้งใจจะพัฒนาโปรแกรมโดยการที่ตัดคำภาษาไทยในหน้าเว็บออกเป็นคำย่อยๆ แล้วจึงนำไปค้นหาในบัญชีดำ แต่จากการศึกษาอัลกอริทึมและโปรแกรมตัดคำภาษาไทยต่างๆ ไม่ว่าจะเป็น ICU หรือ libthai พบว่าไม่สามารถตัดคำได้อย่างถูกต้อง และการที่จะพัฒนาระบบตัดคำขึ้นมาใหม่ก็ไม่ได้เป็นเป้าหมายของโครงการพัฒนาระบบงานนี้ ในระหว่างนั้นก็ได้ทำการศึกษาการทำงานของโปรแกรม DansGuardian โดยละเอียดควบคู่กันไปด้วยซึ่งต้องใช้เวลาพอสมควร เนื่องจากโปรแกรมมีโมดูลย่อยต่างๆ มากมาย และมีการทำงานแยกออกเป็นหลายส่วน จึงพบว่าไม่จำเป็นต้องมีการตัดคำดังได้กล่าวไปแล้วในบทที่ 3 ทำให้สามารถออกแบบระบบตรวจสอบหน้าเว็บภาษาไทยขึ้นมาได้

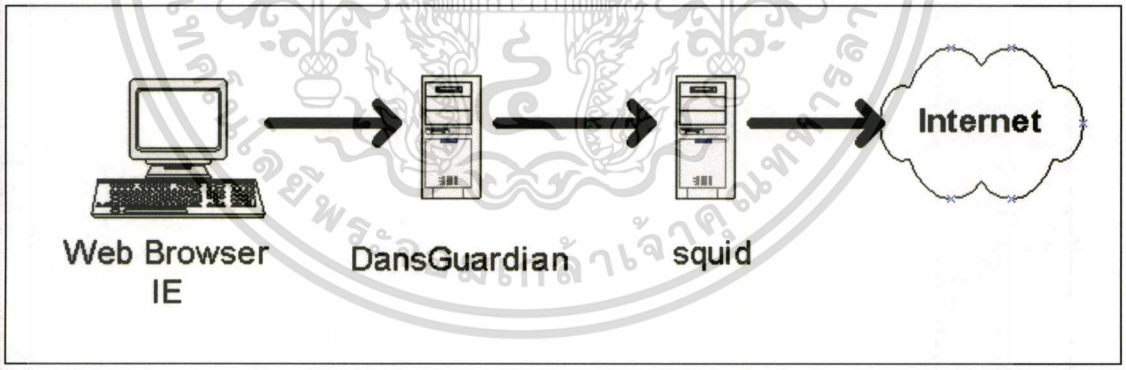
เมื่อได้ศึกษาจนเข้าใจถึงการทำงานของโปรแกรม DansGuardian แล้วจึงได้ออกแบบระบบและเขียนโปรแกรมเพื่อให้ทำงานร่วมกับโปรแกรมเดิมได้ จากนั้นจึงได้ใช้เวลาส่วนใหญ่ในการทดสอบระบบและจัดทำบัญชีดำของวลีที่ไม่เหมาะสม ซึ่งต้องใช้เวลาค่อนข้างมากเนื่องจากต้องกำหนดค่าน้ำหนักของแต่ละวลีให้พอเหมาะ และนำข้อมูลที่ได้จากการทดสอบระบบมาปรับปรุงบัญชีดำให้ถูกต้องมากยิ่งขึ้น

บทที่ 5

การทดสอบโปรแกรม บทสรุป และข้อเสนอแนะ

5.1 การทดสอบการทำงานของโปรแกรม

การทดสอบการทำงานของโปรแกรมนั้นได้ทำโดยติดตั้งโปรแกรมที่ compile เสร็จแล้วลงใน virtual machine ของ VMware พร้อมทั้งติดตั้งโปรแกรม squid เพื่อช่วยประกอบในการทำงาน โดยกำหนดให้โปรแกรมของเรารอรับการติดต่อจาก port 8080 และให้ squid รอรับการติดต่อที่ port 3128 จากนั้นได้ใช้ Internet Explorer (IE) ซึ่งเว็บบราวเซอร์บน Windows 2000 ในการทดลองติดต่อไปยังหน้าเว็บที่มีเนื้อหาไม่เหมาะสมเป็นภาษาไทยต่างๆ โดยให้การติดต่อผ่านการตรวจสอบจากโปรแกรมของเรา ด้วยการกำหนดที่ IE ให้ใช้พรอกซี่เซิร์ฟเวอร์เป็นโปรแกรมของเรา สาเหตุ ที่ต้องใช้โปรแกรม IE เนื่องจากเว็บบราวเซอร์ที่อยู่บน linux ที่ใช้งานอยู่แสดงผลภาษาไทยได้ไม่สมบูรณ์ทำให้อ่านได้ยาก ลักษณะการติดต่อสื่อสารของโปรแกรมจึงเป็นดังรูปที่ 5.1



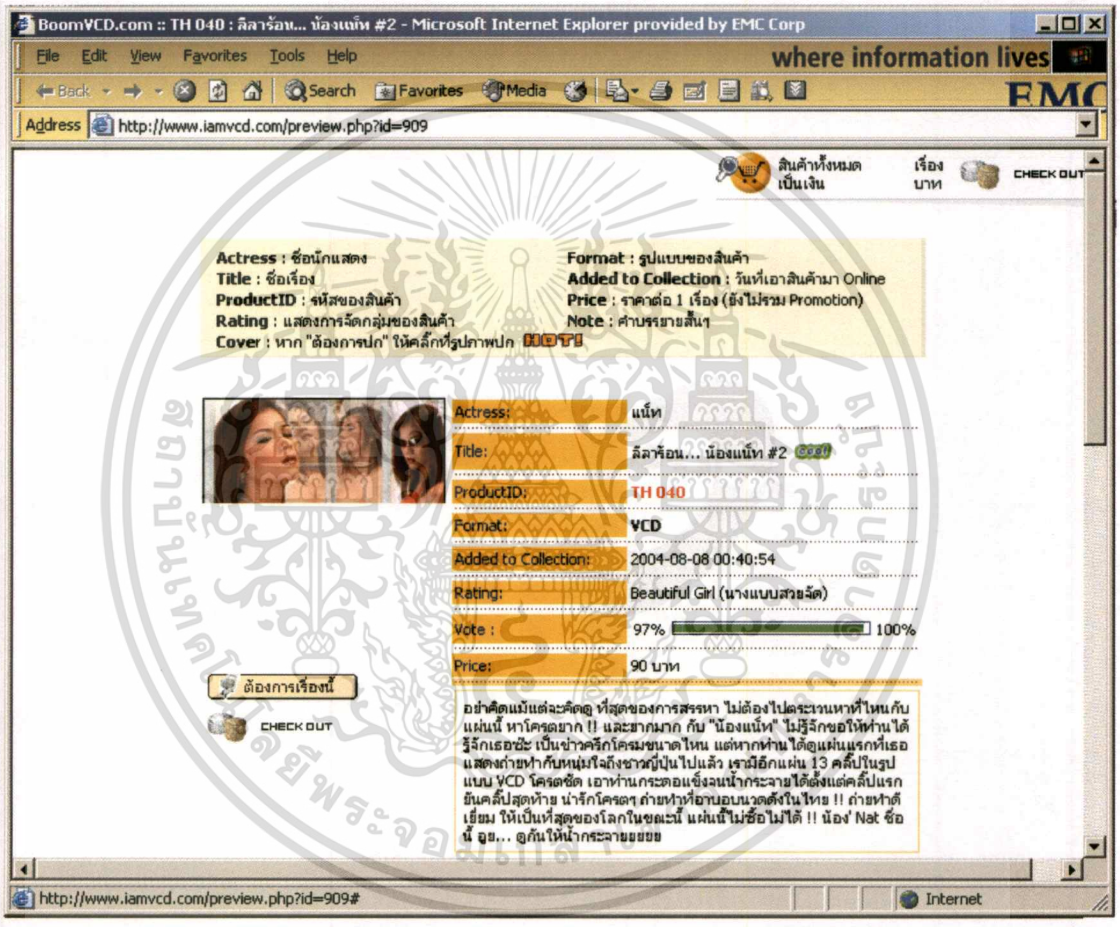
รูปที่ 5.1 ลำดับการติดต่อกันของโปรแกรมต่างๆ

เครื่องที่ใช้ทดสอบมีรายละเอียดดังนี้

- CPU Intel Pentium M 1.8GHz
- RAM 512MB
- OS Windows 2000 workstation
- VMware Workstation 5.0
- LinuxFedora Core 2

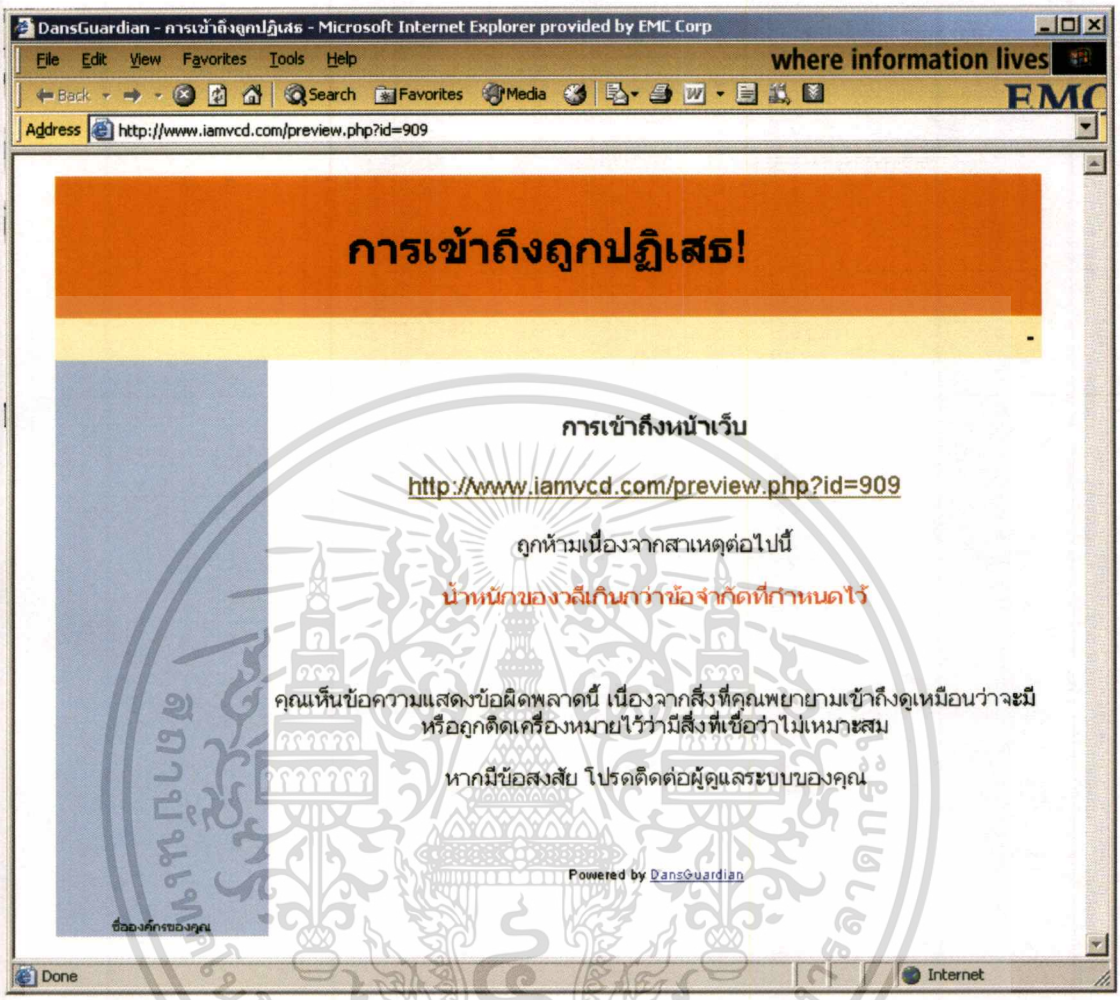
5.2 ขั้นตอนการทดสอบ

1) ทดสอบการใช้งานโดยให้เว็บบราวเซอร์ติดต่อไปยังเว็บไซต์โดยตรงไม่ผ่านโปรแกรม โดยติดต่อไปที่เว็บไซต์ http://www.iamvcd.com/preview.php?id=909 ซึ่งเป็นเว็บที่ขายหนังลามก จะเห็นได้ว่าสามารถติดต่อได้โดยไม่มีปัญหาดังแสดงในรูปที่ 5.2



รูปที่ 5.2 หน้าเว็บที่มีข้อความไม่เหมาะสม

2) ทดสอบการใช้งานโดยผ่านโปรแกรมแต่ไม่กำหนดให้ระบบตรวจสอบภาษาไทยทำงาน ก็จะเห็นได้ว่าสามารถผ่านการตรวจสอบได้เช่นกันถึงแม้ว่าจะมีระบบตรวจสอบก็ตาม
3) ทดสอบการใช้งานโดยผ่านโปรแกรมและกำหนดให้ระบบตรวจสอบภาษาไทยทำงาน คราวนี้จะเห็นได้ว่าระบบของเราสามารถป้องกันการเข้าถึงเว็บไซต์นี้ได้ดังแสดงในรูปที่ 5.3



รูปที่ 5.3 หน้าเว็บที่มีข้อความไม่เหมาะสมถูกห้ามเข้า

5.3 สรุป

จากการพัฒนาระบบป้องกันการเข้าถึงเว็บไซต์ที่ไม่เหมาะสมนี้ ทำให้เราสามารถป้องกันไม่ให้ผู้ใช้งานเข้าไปยังหน้าเว็บที่ไม่เหมาะสมได้ตามความต้องการที่ตั้งไว้ ซึ่งการประยุกต์นำระบบไปใช้งานจริงจะช่วยให้ผู้ใช้งานอินเทอร์เนตมีความสบายใจมากยิ่งขึ้น ว่าจะไม่หลงเข้าไปยังหน้าเว็บที่ไม่ดีเหล่านั้น และหากมีการนำไปใช้งานอย่างกว้างขวางแล้ว ยังจะช่วยป้องกันไม่ให้เยาวชนได้รับข้อมูลข่าวสารที่ไม่เหมาะสมจากอินเทอร์เนตได้เป็นอย่างดี

5.4 ข้อเสนอแนะและแนวทางในการพัฒนาต่อ

หลักจากที่ใช้งานมาได้ระยะหนึ่งได้สังเกตพบว่า ถึงแม้ว่าหน้าเว็บภาษาไทยที่มีปัญหาส่วนใหญ่จะใช้รหัส สมอ. ในการทำหน้าเว็บ แต่ก็ได้พบว่ามีหน้าเว็บบางแห่ง เช่น www.google.co.th มีการเข้ารหัสด้วยรหัส Unicode ซึ่งโปรแกรมนี้ไม่สามารถตรวจสอบได้ จึงอาจเป็นอีกช่องทางหนึ่งที่ผู้จัดทำหน้าเว็บที่ไม่เหมาะสมสามารถหลบเลี่ยงการตรวจสอบได้ ดังนั้นจึงอาจต้องมีการพัฒนาเพิ่มเติมเพื่อให้สามารถแปลงรหัส Unicode กลับมาเป็นรหัส สมอ. เพื่อให้ตรวจสอบได้ดียิ่งขึ้น นอกจากนี้การจัดทำบัญชีดำของวลีที่ไม่เหมาะสมเพิ่มเติม ก็จะช่วยให้ทำการตรวจสอบได้ครอบคลุมมากยิ่งขึ้น และอีกส่วนที่อาจปรับปรุงได้คือคำเตือนต่างๆ และ HTML template ที่เป็นภาษาไทยก็อาจปรับให้มีข้อความที่สละสลวยขึ้น และมีหน้าจอที่สวยงามขึ้นได้

บรรณานุกรม

Barron, D. 2005. **DansGuardian.** [Online]. Available: http://dansguardian.org

Boyer, R.S. and Moore, J.S. 1977. "A Fast String Searching Algorithm." **Communications of the ACM** 20(10): 762-772

Fielding, R. et al. 1999. **Hypertext Transfer Protocol -- HTTP/1.1.** [Online]. Available: http://www.ietf.org/rfc/rfc2616.txt

Marshall, J. 1997. **HTTP Made Really Easy.** [Online]. Available: http://www.jmarshall.com/easy/http/

Oualline, S. 2002. **Practical C++ Programming.** Second Edition. New York: O'Reilly.

Stevens, W. R.1999. **Advanced Programming in the UNIX Environment.** Massachusetts: Addison-Wesley.

ประวัติผู้เขียน

ชื่อ-สกุล	นายอภิศิษฏ์ ทนันชัย
วัน-เดือน-ปี เกิด	22 พฤศจิกายน 2516
ที่อยู่	40/46 หมู่ 6 ต.คูคต อ.ลำลูกกา จ.ปทุมธานี 12130
อีเมล์	tapisit@yahoo.com
ประวัติการศึกษา	- ชั้นประถมศึกษา โรงเรียนวัดเมธังกราวาสฯ - ชั้นมัธยมศึกษา โรงเรียนพิริยาลัยฯ - ปริญญาตรี วิศวกรรมศาสตร์บัณฑิต ภาควิชาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยเกษตรศาสตร์
ที่ทำงานปัจจุบัน	บริษัท อีเอ็มซี อินฟอร์เมชั่น ซิสเท็มส์ (ประเทศไทย) จำกัด