เครื่องมือเก็บหลักฐานสำหรับการรักษาความปลอดภัยทางคอมพิวเตอร์

COMPUTER SECURITY FORENSIC TOOLS

ศรัทธา ชลบุษย์

สุรชัย คงสุข

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2551

ปริญญานิพนธ์ปีการศึกษา 2551

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง เครื่องมือเก็บหลักฐานสำหรับการรักษาความปลอดภัยทางคอมพิวเตอร์

Computer Security Forensic Tools

ผู้จัดทำ

1. นายศรัทธา ชลบุษย์ รหัสนักศึกษา 49015304

2. นายสุรชัย คงสุข รหัสนักศึกษา 49015311

_____ อาจารย์ที่ปรึกษา

(อาจารย์อัครเดช วัชระภูพงษ์)

_____ อาจารย์ที่ปรึกษา

(ผู้ช่วยศาสตราจารย์ธนา หงษ์สุวรรณ)

_____ อาจารย์ที่ปรึกษา

(อาจารย์ธนัญชัย ตรีภาค)

เครื่องมือเก็บหลักฐานสำหรับการรักษาความปลอดภัยทางคอมพิวเตอร์

นาย ศรัทธา	ชลบุษย์	
นาย สุรชัย	คงสุข	
อาจารย์ อัครเดช	วัชระภูพงษ์	อาจารย์ที่ปรึกษา
ผู้ช่วยศาสตราจารย์ ธนา	หงษ์สุวรรณ	อาจารย์ที่ปรึกร่วม
อาจารย์ ธนัญชัย	ตรีภาค	อาจารย์ที่ปรึกร่วม

บทคัดย่อ

ปัจจุบันการละเมิดระบบรักษาความปลอดภัยทางคอมพิวเตอร์และเครือข่ายเกิดขึ้นบ่อยครั้งแม้มีมาตรการป้องกันเป็นอย่างดี ทั้งที่เกิดจากความตั้งใจและไม่ตั้งใจ ทั้งจากตัวผู้ใช้เองและบุคคลที่มาบุกรุกหรือแอบใช้งานผ่านช่องทางต่างๆ ซึ่งเมื่อเกิดเหตุขึ้นนอกจากการตอบสนองการละเมิดความปลอดภัยด้วยการระงับเหตุให้ได้ทันท่วงทีแล้ว ยังควรมีการสืบค้นเพื่อตามเก็บหลักฐานเพื่อศึกษาย้อนรอย หรือดำเนินการทางกฎหมายกับผู้ละเมิด ทั้งนี้ต้องเป็นไปตามแนวทาง พรบ.ว่าด้วยการกระทำความผิดเกี่ยวกับคอมพิวเตอร์ พ.ศ. 2550 และกฎระเบียบที่เกี่ยวข้อง

การค้นหาและเก็บหลักฐานทางดิจิตอลที่อยู่ในอุปกรณ์คอมพิวเตอร์ หลักฐานทั้งหมดนี้จะถูกนำมาวิเคราะห์ว่า เกิดขึ้นเมื่อไหร่ จากอะไร ขณะใช้งานทำอะไรกับข้อมูลอยู่และถูกใช้โดยใคร โดยทำการเก็บหลักฐานมาพิจารณาและส่งทำการวิเคราะห์เพื่อนำเสนอหลักฐานในชั้นศาล หลักฐานที่เป็นดิจิตอลนี้มีความละเอียดอ่อนมาก เนื่องจากสามารถถูกทำลายหรือได้รับความเสียหายจากการไม่ระมัดระวังได้ อีกทั้งยังสามารถที่จะซ่อนแก้ไขหรือเปลี่ยนแปลงข้อมูล เพื่อทำให้หลักฐานนั้นมีการบิดเบือนไปต่างจากหลักฐานที่เป็นอยู่เดิมได้

ดังนั้นเพื่อเป็นการป้องกันหากเกิดการละเมิดความปลอดภัยขึ้น จึงต้องมีการที่สืบค้นหลักฐานโดยใช้ความสามารถในการคัดลอก (Cloning) กู้คืน (Undelete & Unformat) และแกะรหัสผ่าน (Zip/Rar Password Cracking) เพื่อนำข้อมูลจากสื่อต่างๆ เช่น ฮาร์ดดิสก์ แฟลชเมโมรีต่างๆ โดยเน้นไปในลักษณะของการทำงานแบบออฟไลน์กับเครื่องหรือสื่อเป้าหมายที่ต้องการสืบค้น

Computer Security Forensic Tools

Mr. Satta Chonlabut

Mr. Surachai Kongsook

Mr. Akkradach Watcharapupong Advisor

Asst.Prof. Thana Hongsuwan Co-Advisor

Mr. Thananchai Treepak Co-Advisor

Academic Year 2008

ABSTRACT

Nowadays, breaking the computer and network security, though with strict policy fully protected, is a commonplace in our IT society, whether by accident or on purpose, from the users misconduct or the outside attackers who trespass or sneak to exploit resources from open vulnerabilities. Once the threats occurred,

with security measures urgently responded on, there should be a tracing action to track back evidences and/or execute on the trespassors according to Thailand Computer Crime Law (2007).

The digital evidence tracing and collection in any computer devices are used for deep analysis as to when it has happened, from what causes, on what activities and by whom. Once evidences collected, there follows the evidence authentication process and evidence analysis with the objective for the court presentation. Digital evidence is highly vulnerable since it can be easily destroyed and damaged by careless handling. Moreover, the data can be hidden, edited or modified by the malicious intention to distort some particular truths.

To prevent from unlawful security transgression, multiple techniques are applied for evidence tracing and collection process, such as Cloning, Undeleting & Unformatting, Zip/Rar Password Cracking on the digital media like Hard Disks or any form of Flash Memories towards the offline operation on a targeted machine or media under investigation.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้ได้รับคำแนะนำและให้คำปรึกษาเกี่ยวกับการวิจัยพัฒนาและการค้นคว้าจากอาจารย์อัครเดช วัชระภูพงษ์ อาจารย์ผู้ควบคุมปริญญานิพนธ์ ผู้ช่วยศาสตราจารย์ ธนาหงษ์สุวรรณ และอาจารย์ ธนัญชัย ตรีภาค ผู้ควบคุมปริญญานิพนธ์ร่วม ที่ได้ช่วยดูแลให้คำปรึกษาด้วยดีมาโดยตลอด ขอบคุณห้องวิจัย ISAG คณะภาควิชาวิศวกรรมพิวเตอร์ที่ได้สนับสนุนในส่วนของห้องปฏิบัติการวิจัย รวมถึงเครื่องมือ หนังสือและความรู้ที่ได้รับการถ่ายทอดเพื่อการวิจัยในครั้งนี้

ขอขอบคุณเพื่อนๆ พี่ๆ น้องๆ ในภาควิชาวิศวกรรมพิวเตอร์ ที่ช่วยเป็นกำลังใจให้การช่วยเหลือสนับสนุนและแบ่งปันความรู้ที่มีมาโดยตลอด

สุดท้ายนี้กลุ่มของข้าพเจ้าขอกราบขอบพระคุณ บิดามารดา และครอบครัวที่เป็นกำลังใจ และให้การสนับสนุนในทุกๆเรื่องทำให้กลุ่มของข้าพเจ้าพัฒนาวิจัยจนปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงด้วยดี

คุณค่าและคุณประโยชน์อันพึงมาจากผลการวิจัยปริญญานิพนธ์ฉบับนี้กลุ่มของข้าพเจ้าขอมอบแด่ผู้มีพระคุณทุกท่าน

นาย ศรัทธา ชลบุษย์

นาย สุรชัย คงสุข

สารบัญ

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญตาราง

ตารางที่ **หน้า**

สารบัญรูป

รูปที่ **หน้า**

สารบัญรูป(ต่อ)

รูปที่ **หน้า**

บทที่ 1
บทนำ

1.1 ความสำคัญและที่มา

ปัจจุบันการละเมิดระบบรักษาความปลอดภัยทางคอมพิวเตอร์และเครือข่ายเกิดขึ้นบ่อยครั้ง แม้มีมาตรการป้องกันเป็นอย่างดี ทั้งที่เกิดจากความตั้งใจและไม่ตั้งใจ ทั้งจากตัวผู้ใช้เองและบุคคลที่มาบุกรุกหรือแอบใช้งานผ่านช่องทางต่างๆ ซึ่งเมื่อเกิดเหตุขึ้นนอกจากการตอบสนองการละเมิดความปลอดภัยด้วยการระงับเหตุให้ได้ทันท่วงทีแล้ว ยังควรมีการสืบค้นเพื่อตามเก็บหลักฐานเพื่อศึกษาย้อนรอย หรือดำเนินการทางกฎหมายกับผู้ละเมิด ทั้งนี้ต้องเป็นไปตามแนวทาง พรบ.ว่าด้วยการกระทำความผิดเกี่ยวกับคอมพิวเตอร์ พ.ศ. 2550 และกฎระเบียบที่เกี่ยวข้อง

การค้นหา และเก็บหลักฐานทางดิจิตอลที่อยู่ในอุปกรณ์คอมพิวเตอร์ เช่น ไฟล์ที่อยู่ใน พีซี โน้ตบุ๊ก เป็นต้น หรือหลักฐานดิจิตอลที่ถูกสร้างจากระบบคอมพิวเตอร์ เช่น บันทึกการใช้งานข้อมูลของการใช้อินเทอร์เน็ตเป็นต้น ซึ่งหลักฐานทั้งหมดนี้จะถูกนำมาวิเคราะห์ว่าหลักฐานนี้เกิดขึ้นเมื่อไหร่ จากอะไร ตอนนี้ใช้ทำอะไร และถูกใช้โดยใคร โดยกระบวนการสืบค้นจะประกอบไปได้ด้วย การเก็บหลักฐาน การพิสูจน์ความถูกต้องของหลักฐาน และการวิเคราะห์หลักฐานเพื่อนำเสนอในชั้นศาล หลักฐานที่เป็นดิจิตอลนี้มีความละเอียดอ่อนมาก เพราะสามารถถูกทำลาย หรือเกิดความเสียหายโดยความไม่ระมัดระวังได้ อีกทั้งยังสามารถที่จะซ่อน หรือเปลี่ยนแปลงข้อมูลเพื่อทำให้หลักฐานนั้นมีการบิดเบือนไป ต่างจากหลักฐานที่เป็นสิ่งที่ปรากฏชัดอย่างเช่น ลายนิ้วมือ ซึ่งไม่สามารถที่จะทำการปลอมหรือเปลี่ยนแปลงได้ ซึ่งหากหลักฐานมีการบิดเบือนการสืบสวนก็จะไปผิดทางด้วย

กระบวนการที่จะสืบค้นหรือย้อนรอยเพื่อตามหาหลักฐานการกระทำผิดนั้นสามารถทำได้หลากหลายวิธีส่วนหนึ่งคือกระบวนการสืบหาไฟล์ข้อมูลที่ผู้กระทำผิดอาจจะต้องการลบเพื่อปกปิดร่องรอยหรือหลักฐานการกระทำความผิด ซึ่งโดยปกติการจัดการไฟล์ของระบบปฏิบัติการนั้นตัวข้อมูลต่างๆที่ผู้ใช้งานใช้อยู่จะถูกจัดเก็บไว้บนสื่อซึ่งสามารถทำการ สร้าง ลบ แก้ไขหรือดัดแปลงได้ ดังนั้นเมื่อมีความพยายามที่จะทำลาย ไฟล์ข้อมูลซึ่งอาจจะเป็นไฟล์ที่ผิดกฎหมายหรือไฟล์ที่

อาจจะจะก่อให้เกิดความผิดพลาดหรือก่อให้เกิดความเสียหายแก่ระบบ บุคคล หรือหน่วยงานที่เกี่ยวข้อง หากไฟล์เหล่านั้นได้รับการเผยแพร่หรือสำเนาแจกจ่ายออกไป

ดังนั้นเมื่อเกิดการละเมิดความปลอดภัยขึ้น จึงต้องมีการที่จะสืบค้นหลักฐาน โดยใช้ความสามารถในการคัดลอก (Cloning) กู้คืน (Undelete & Unformat) และแกะรหัสผ่าน (Zip/Rar Password Cracking) ข้อมูลจากสื่อเก็บข้อมูลต่างๆ เช่น ฮาร์ดดิสก์ แฟลชเมโมรีต่างๆ โดยเน้นไปในลักษณะของการทำงานแบบออฟไลน์กับเครื่องหรือสื่อเป้าหมายที่ต้องการสืบค้น

1.2 วัตถุประสงค์

1.2.1 เพื่อศึกษาแนวทางการตอบสนองเมื่อเกิดการละเมิดความปลอดภัย

1.2.2 เพื่อศึกษาวิธีปกปิดพฤติกรรมของผู้บุกรุกระบบคอมพิวเตอร์

1.2.3 เพื่อศึกษามาตรการเก็บหลักฐานสำหรับการรักษาความปลอดภัยทางคอมพิวเตอร์

1.2.4 เพื่อสร้างต้นแบบเครื่องมือเก็บหลักฐานสำหรับการรักษาความปลอดภัยทางคอมพิวเตอร์

1.3 ขอบเขตของงานวิจัย

1.3.1 พัฒนา Live CD เพื่อใช้รันบนเครื่องเป้าหมายที่ต้องการ โดยไม่ต้องทำการถอดหรือเคลื่อนย้ายอุปกรณ์ โดยไม่ขึ้นกับระบบปฏิบัติการของเครื่องเป้าหมาย

1.3.2 ใช้ความสามารถของโปรแกรมที่บรรจุใน Live CD ที่พัฒนาขึ้นเพื่อทำการสำเนาข้อมูลจากเครื่องเป้าหมาย ไปยังสื่ออื่นเพื่อใช้สำหรับการตรวจสอบหรือสำรองข้อมูล

1.3.3 ใช้ความสามารถของโปรแกรมที่บรรจุใน Live CD ที่พัฒนาขึ้นเพื่อกู้ข้อมูลที่สูญหายทั้งที่เกิดจากการลบ ปิดบัง ทั้ง ฟอร์แมต(Format)หรือ ลบ(Delete) ทั้งแบบตั้งใจและไม่ตั้งใจจากเครื่องเป้าหมาย

1.3.4 สร้างโปรแกรมต้นแบบเพื่อถอดรหัสไฟล์ Zip/Rar

1.4 ขั้นตอนการดำเนินงาน

1.4.1 ศึกษาแนวทางในการพัฒนา Live CD เพื่อใช้รันบนเครื่องเป้าหมาย

1.4.2 ศึกษาโครงสร้างของฮาร์ดดิสก์ และระบบไฟล์แบบ FAT

1.4.3 ศึกษาแนวทางและออกแบบในการพัฒนาโปรแกรมเพื่อกู้ข้อมูล(undelete/unformat)

1.4.4 ศึกษาการถอดรหัสไฟล์ที่มีการตั้งรหัส และออกแบบโปรแกรมเพื่อใช้ในการถอดรหัส ไฟล์ Zip/Rar

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1.5.1 เมื่อหลักฐานบางส่วนได้ถูกผู้ไม่ประสงค์ดี ลบหรือทำลายหลักฐาน เราสามารถกู้หลักฐานส่วนนั้นเพื่อนำมาใช้ประกอบกับการดำเนินคดีได้

1.5.2 การนำเทคโนโลยีการถอดรหัสข้อมูลมาใช้ เป็นการเพิ่มประสิทธิภาพการสืบค้นข้อมูล กล่าวคือ ผู้ไม่ประสงค์ดีบางครั้งต้องการจงใจที่จะเข้ารหัสข้อมูลไว้ เพราะต้องการที่จะปกปิดหลักฐาน ซึ่งการถอดรหัสข้อมูลในส่วนนี้จึงช่วยให้สามารถที่จะนำหลักฐานที่ถูกปกปิดไว้ ออกมาใช้งานได้

1.5.3 การนำ Live CD เพื่อนำมาเข้าถึงข้อมูลหรือหลักฐานของเครื่องเป้าหมาย ทำให้เราไม่จำเป็นต้องสนใจระบบปฏิบัติการของเครื่องเป้าหมายเลย ทำให้สะดวกแก่การเก็บหลักฐานมากขึ้น

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 หลักการพื้นฐานของ LiveCD

ในการทำความเข้าใจในหลักการทำงานของ LiveCD นั้นในส่วนแรกที่ต้องทำความเข้าใจคือ LiveCD คือศัพท์ที่ถูกนำมาเรียกการทำงานของระบบปฏิบัติการที่สามารถทำงานจากแผ่นซีดีได้โดยในช่วงแรกเริ่มนั้นพัฒนาขึ้นเพื่อใช้กับลินุกส์(linux) โดยไม่จำเป็นต้องอาศัยความสามารถฮาร์ดดิสต์แต่อย่างใด หลักการคร่าวๆของ LiveCD ก็คือระบบปฏิบัติการลินุกซ์ที่ถูกจำลองไว้บนซีดีในรูปแบบ SquashFS ซึ่ง SquashFS คือชื่อเรียกระบบซิสเต็มไฟล์ชนิดหนึ่ง(ที่เริ่มมีใช้เฉพาะลินุกส์) ระบบไฟล์ดังกล่าวจะถูกบีบย่อในอัตราที่สูงโดยประมาณถึง 3 เท่าตัวของเนื้อที่ใช้งานจริง ซึ่งระบบไฟล์ลินุกซ์โดยปกติจะใช้เนื้อที่ใน ฮาร์ดดิสก์(Hardisk) โดยประมาณ 2 GB แต่จะถูกบีบย่อให้เหลือไม่ถึง 700 เมกกะไบต์ หรือให้มีขนาดเทียบเท่าแผ่นซีดี หนึ่งแผ่นเท่านั้น แต่เนื่องจากระบบไฟล์ชนิดนี้ต้อทำงานบนแผ่นซีดี ดังนั้นย่อมมีข้อจำกัดในการใช้งาน เนื่องจากสามารถทำการอ่านเพื่อใช้งานได้อย่างเดียวเท่านั้น ไม่สามารถบันทึกจัดเก็บค่าหรือทำการเปลี่ยนแปลงไฟล์ใดๆคืนกลับไปได้ แต่ข้อด้อยดังกล่าวจะถูกนำมาประยุกต์ใช้งานกับระบบเมมโมรี่ไฟล์ซิสเต็ม (RamFS) ซึ่งเป็นระบบซิสเต็มไฟล์ที่ทำงานบนเมมโมรี่ในรูปแบบไดนามิก (หลักการถ่ายเทข้อมูลระว่างสื่อจัดเก็บและหน่วยความจำซึ่งจะทำการถ่ายเทข้อมูลไปใช้งานเมื่อจำเป็นเท่านั้น)

2.1.1 แนวคิดของ SquashFS

แรกเริ่มที่มีการสร้างอุปกรณ์และมีความจำเป็นที่ต้องใช้งานระบบสมองกลฝังตัว ซึ่งมีการใช้งานระบบปฏิบัติการลินุกซ์นั้น ในทุกๆไบต์ของสื่อเก็บข้อมูลเช่น ฟลอปปีดิสก์(Floppy Disk) , แฟลชดิสก์(Flash Disk) , อื่น ๆ นั้นมีความสำคัญมาก การบีบอัดข้อมูลจึงถือเป็นเรื่องสำคัญมาก ไฟล์ข้อมูลที่มีขนาดใหญ่จนถึงข้อมูลส่วนบุคคลและไฟล์ระบบนั้นก็ถือเป็นจุดหนึ่งที่มีความสำคัญที่ควรจะมีการบีบอัดเพื่อพื้นที่ในการจัดเก็บไฟล์

SquashFS คือกระบวนการที่เข้ามาช่วยในส่วนนี้ สำหรับไฟล์ระบบที่สามารถอ่านได้อย่างเดียวนั้นจะทำการบีบอัดเป็นไฟล์ระบบไฟล์เดียว จากนั้นทำการเขียนอุปกรณ์ พาทีชั่น(Partition) หรือไฟล์ธรรมดาๆจากนั้นทำการ เม้า(mount) โดยตรงในกรณีถ้าเป็นอุปกรณ์ หรือใช้ในลักษณะ

ของ ลุกอัพ(Loopback) ในกรณีที่เป็นไฟล์ ซึ่งการแยกเป็นแต่ละโมดูลของระบบ SquashFS นั้นเป็นจุดประสงค์หนึ่งที่ช่วยให้เกิดความยืดหยุ่นและเพิ่มประสิทธิภาพในเรื่องของความเร็วเมื่อทำการแตกไฟล์ออกมาโดยภาพรวมของระบบ SquashFS เป็นดังนี้

- ส่วนข้อมูล ไอโหนด(Inodes) และ ไดเรกทอรี(Directory) ถูกบีบบัดไว้
- ขนาดไฟล์สามารถมีขนาดได้สูงสุด 2^64 ไบต์
- ไอโหนด และ ไดเรกทอรี ถูกบีบอัดในอัตราที่สูงและถูกจัดเก็บแบ่งแยกกัน โดยในแต่ละ ไอโหนดจะมีขนาดโดยเฉลี่ย 8 ไบต์
- SquashFS สามารถใช้ขนาดของ Block ได้สูงสุด 64 กิโลไบต์ สำหรับการบีบอัดแบบ 2.x และสูงสุดที่ 1เมกะไบต์ สำหรับการบีบอัดแบบ 3.x โดยปกติขนาดของบล็อก(Block) จะอยู่ที่ 128 กิโลไบต์
- ที่การบีบอัดขนาด 2.x ตามคอนเซ็บของแฟลกเมน บล็อก(Fragment blocks) นั้นสามารถที่จะเพิ่มไฟล์ขนาดเล็กๆเข้าไปในบล็อกเดี่ยวบล็อกเดียวได้
- ไฟล์เดียวกันที่ซ้ำกันจะถูกลบออกและใช้แค่ไฟล์เดียวเท่านั้น
- รองรับสถาปัตยกรรมแบบ บิ๊กเอนเดียน(Big endian) และ ลิตเติล เอนเดียน(Little endian) โดย SquashFS สามารถที่จะเม้า ไฟล์ระบบที่สร้างบนเครื่องที่แตกต่างกันได้

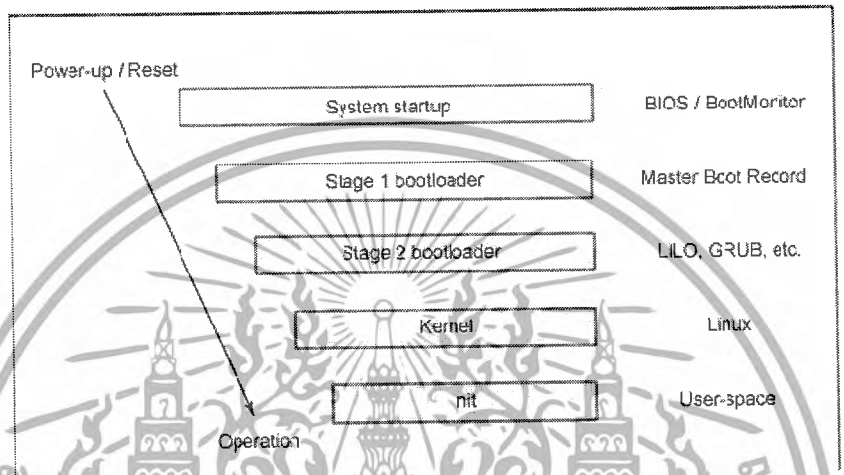
2.1.2 แนวคิดของ RamFS

RamFS เป็นส่วนขยายของ Linux's disk cacheing mechanisms ซึ่งจะปรับเปลี่ยนไปตามขนาดของหน่วยความจำแต่ละเครื่อง โดยปกติแล้วไฟล์จะถูกดึงเข้ามาอยู่ในหน่วยความจำหลักอยู่แล้ว แต่ในบางครั้งก็อาจจะไม่มีอยู่ในหน่วยความจำหลักแต่จะไปอยู่ในหน่วยความจำเสมือนแทน เนื่องจากระบบอาจจะต้องการที่จะให้หน่วยความจำหลักในการทำงาน ดังนั้นข้อมูลจึงถูกย้ายไปมาระหว่างหน่วยเก็บข้อมูลและหน่วยความจำเสมือนเมื่อระบบต้องการเรียกใช้งานก็สามารถเรียกใช้งานจากหน่วยความจำเสมือนได้ทันทีซึ่งเป็นการประหยัดเวลาในการเข้าถึงข้อมูล

เนื่องด้วย RamFS ไม่มีหน้าที่ในการจัดเก็บแบบถาวร แต่ไฟล์ข้อมูลจะถูกแทนลง RamFS ที่ได้ถูกจองเอาไว้ในลักษณะเดียวกับ เพจแคช(page cache) แต่จะไม่มีการเขียนไฟล์ข้อมูลลงไปจริงๆ หมายความว่ามันจะมีส่วนของข้อมูลตลอดเวลาอยู่แล้ว และจะไม่สามารถทำการเคลียร์ออกไปได้

โดยรวมแล้ว RamFS นั้นมีขนาดเล็กเนื่องจากทำงานโดยใช้โครงสร้างพื้นฐานของ ลินุกซ์ แคชชิ่ง พูดง่ายๆว่าเหมือนการดึงเข้ามาใช้งานเป็นส่วนนึงของระบบ เนื่องจากว่า RamFS ไม่ใช่ส่วนของตัวเลือกที่สามารถที่จะนำออกไปได้

2.1.3 กระบวนการบู๊ต(Boot) พื้นฐานของระบบของลินุกซ์



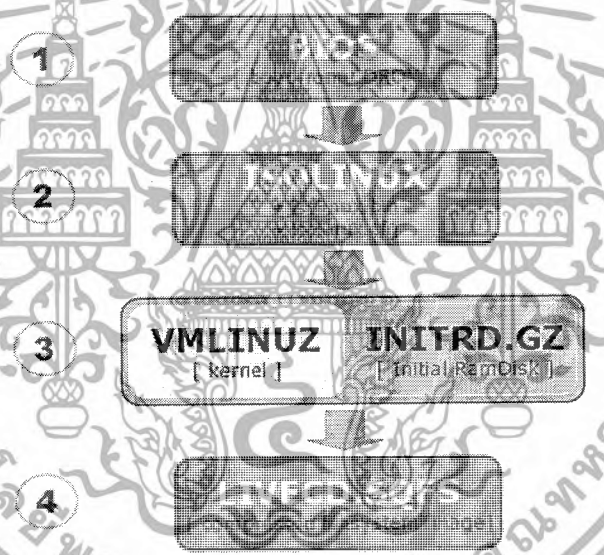
รูปที่ 2.1 แสดงกระบวนการบู๊ต(boot) ของระบบปฏิบัติการลินุกซ์

- System startup ในขั้นแรกนั้นเมื่อทำการเปิดเครื่องระบบแรกที่ทำงานนั้นคือส่วนของ ไบออส(Bios) ซึ่งจะทำการตรวจสอบอุปกรณ์เชื่อมต่อทั้งหมดที่ติดตั้งอยู่เมื่อทุกอย่างผ่านหมดแล้วจึงจะทำการข้ามขั้นไปหาส่วนของการโหลดระบบจากสื่อจัดเก็บ
- Stage 1 boot loader นั้นคือส่วนของสื่อจัดเก็บข้อมูลที่จะติดตั้งข้อมูลที่จะแจ้งให้กับระบบพื้นฐานทราบว่ามีระบบปฏิบัติการใดติดตั้งอยู่บ้างและติดตั้งอยู่ ณ ส่วนไหนของสื่อเก็บข้อมูล
- Stage 2 boot loader ในส่วนนี้ GRUB เป็นบู๊ตโหลดเดอ ซึ่งพัฒนาโดย GNU ซึ่งมีความสามารถในการทำ MultiBoot Loader โดยการปรับเปลี่ยนค่าของ อิเมจ เคอร์เนล ได้ หรือสามารถทำการส่งค่าพารามิเตอร์ ไปให้กับเคอร์เนล เป็นต้น โดยเมื่อคอมพิวเตอร์เริ่มทำงานจะทำการโหลด GRUB เข้ามาเป็นลำดับแรก และ GRUB นี้จะทำการส่งข้อมูลและการควบคุม (Control) ไปให้ส่วนของเคอร์เนล ของระบบปฏิบัติการ (LILO เป็นหนึ่งในโปรแกรมประเภท บู๊ตโหลดเดอ)
- Load kernel ในส่วนนี้เป็นขั้นตอนการบู๊ตเคอร์เนล จะมีหน้าที่ในการมองหาอุปกรณ์ในเครื่อง หากเจอจะทำการ โหลดไดร์เวอ(driver) ของอุปกรณ์นั้นๆ และโหลดไฟล์ซิสเตมมาทำงาน

- Load Initrd ในส่วนของการโหลด initrd นั้น จะเป็นการ โหลดโมดูล ต่างๆ เพิ่มเติม ตาม จากนั้นจึงทำการโหลดบริการ(service) เพื่อสร้าง โพรเซส(process) มาใช้งาน ต่างๆตามความต้องการของผู้ใช้ โดยการโหลด /etc/rc.d ตามที่ เจ้าของระบบนั้นตั้ง ไว้ ตัวอย่างบริการง่ายๆที่เรารู้จักกันดี เช่น HTTP Service (บริการสำหรับโพรโท คอล http) หรือ Mysql Service (บริการสำหรับ Mysql Database) เป็นต้น

2.1.4 กระบวนการบู๊ตระบบของ LiveCD

การบู๊ตระบบการทำงานของ LiveCD จะแตกต่างจากการบู๊ตลินุกซ์ปกติเล็กน้อยคือมีส่วนของการ เม้าระบบและ ยูเนียนระบบ เนื่องจากว่าลินุกซ์ ที่อยู่บนซีดี นั้นจะมีข้อจำกัดที่ว่าซีดี นั้นไม่สามารถเขียนข้อมูลลงบนไฟล์ได้โดยตรง จึงต้องมีการทำ chroot system เพื่อให้เสมือนว่ามีการโหลดระบบปฏิบัติการอีกระบบหนึ่งให้ทำงาน โดยมีลำดับขั้นตอนการทำงานดังนี้



รูปที่ 2.2 แสดงกระบวนการบู๊ตระบบของลินุกซ์

- System startup ในขั้นแรกนั้นเมื่อทำการเปิดเครื่องระบบแรกที่ทำงานนั้นคือส่วนของไบออส ซึ่งจะทำการตรวจสอบอุปกรณ์เชื่อมต่อทั้งหมดที่ติดตั้งอยู่เมื่อทุกอย่างผ่านหมดแล้วจึงจะทำการข้ามขั้นไปหาส่วนของการโหลดระบบจากสื่อจัดเก็บ โดยจะทำการโหลดข้อมูลที่แจ้งให้กับระบบพื้นฐานทราบว่ามีระบบปฏิบัติการใดติดตั้งอยู่บ้างและติดตั้งอยู่ ณ ส่วนไหนของสื่อเก็บข้อมูล
- ในส่วนนี้เป็นส่วนบู๊ตโหลดเดอจะทำการโหลด initrd.gz ขึ้นมาเพื่อส่งการส่งข้อมูลและการควบคุมให้กับเคอร์เนลระบบปฏิบัติการ

- โหลดเคอร์เนล ในส่วนนี้นั้นจะทำงานโดยแบ่งออกเป็น 2 ขั้นตอนคือน โหลดเคอร์เนล ของระบบขึ้นมาเพื่อที่จะสามารถจัดการกับทรัพยากรของเครื่องนั้นๆได้ หลังจากนั้นจึงเปลี่ยนเป็นเคอร์เนลของลินุกซ์ ที่ได้จัดเตรียมไว้ในแผ่นตามไฟล์ initrd.gz โดยขั้นตอนนี้ทำการเม้าอุปกรณ์และเม้าระบบ เพื่อใช้เป็นพื้นที่ในการทำงานของ LiveCD และมีการทำยูเนียนระบบ เพื่อเชื่อม อุปกรณ์ดังกล่าวทั้งหมดทำงานร่วมกัน
- ขั้นตอนสุดท้ายคือกระบวนการ Chroot ไปยังลินุกส์ ที่อยู่ในลักษณะไฟล์ที่ถูกบีบอัดไว้เพื่อใช้เพื่อเข้าใช้สภาพแวดล้อมของลินุกส์ ที่ได้จัดเตรียมไว้จากนั้นจึงปล่อยให้ระบบทำการโหลดบริการ ที่ได้ทำการติดตั้งไว้ขึ้นมาใช้งาน

2.2 ระบบไฟล์

ระบบไฟล์คือโครงสร้างการจัดเก็บข้อมูลในฮาร์ดดิสก์ เพื่อให้ระบบปฏิบัติการสามารถอ่านเขียนใช้งานไฟล์ที่ต้องการได้อย่างมีประสิทธิภาพ และควบคุมสิทธิในการเข้าใช้งานของผู้ใช้ เช่น ในระบบปฏิบัติการดอส (Disk Operating System) ที่มีการใช้งานมาเป็นระยะเวลายาวนานมีระบบไฟล์ที่เรียกว่า FAT (File Allocation Table) ซึ่งมีทั้งแบบ 12 และแบบ 16 บิต และสำหรับระบบไฟล์ในไมโครซอฟต์วินโดวส์ 95 ของไมโครซอฟต์และ OS/2 ของไอบีเอ็มสนับสนุนระบบไฟล์ทั้งแบบ FAT12, FAT16 และ FAT32 แต่ระบบปฏิบัติการทั้งสองแบบเป็นระบบปฏิบัติการแบบผู้ใช้เดียว ทำให้ระบบไฟล์ไม่มีส่วนของการควบคุมสิทธิการเข้าใช้งาน แต่ระบบปฏิบัติการใหม่ๆเช่น ไมโครซอฟต์วินโดวส์ เอ็นที จะใช้ระบบไฟล์ที่เรียกว่า NTFS (NT File System) หรือระบบปฏิบัติการ IBM OS/2 จะใช้ระบบไฟล์ที่เรียกว่า HPFS (High Performance File System) ซึ่งทั้งสองระบบจะมีส่วนที่ควบคุมสิทธิการเข้าใช้ของผู้ใช้ มีส่วนสนับสนุนการเข้าจากผู้ใช้หลายคน

ตารางที่ 2.1 ระบบไฟล์ของระบบปฏิบัติการชนิดต่างๆ

ระบบปฏิบัติการ	ระบบไฟล์
DOS	FAT12, FAT16
MS Windows 95	FAT12, FAT16
MS Windows 95, MS Windows 98, MS Windows ME	FAT12, FAT16, FAT32
IBM OS/2	FAT12, FAT16, HPFS
Linux	FAT12, FAT16, FAT32, HPFS, NTFS, Ext1, Ext2, Ext3, UFS, VFAT, Minix, ISOFS, HFS, AFFS, ADFS, SYSV

2.2.1 ระบบชื่อไฟล์ในระบบลินุกซ์

ระบบการตั้งชื่อไฟล์ในระบบปฏิบัติการระบบลินุกซ์ นั้นแตกต่างจากดอสคือ ดอสจะให้มีการตั้งชื่อไฟล์ด้วยชื่อขนาด 8 ตัวอักษรและสกุลอีก 3 ตัวอักษร ส่วนระบบไฟล์ของระบบลินุกซ์นั้นในรุ่นแรกๆนั้นจะใช้ระบบไฟล์ของ Minix (Minix File system) ซึ่งเป็นระบบไฟล์ที่ถูกใช้ในระบบปฏิบัติการ Minix นี้จะมีขีดความสามารถจำกัดในการเก็บชื่อไฟล์ได้สูงสุดเพียง 14 ตัวและมีข้อจำกัดมากมายในระบบไฟล์ต่อมาได้มีการปรับปรุงและพัฒนาเพิ่มเพื่อให้สามารถเก็บชื่อไฟล์ที่มีขนาดเพิ่มขึ้นได้ถึง 30 ตัวอักษรสำหรับลินุกซ์ ในปัจจุบันนี้ใช้ระบบไฟล์ที่เรียกว่า Ext2 (Extended Files System 2) ซึ่งสามารถตั้งชื่อไฟล์ได้ถึง 255 ตัวอักษร

การตั้งชื่อไฟล์ในระบบลินุกซ์ นั้นสามารถใช้ตัวอักษร ตัวเลข ขีดเส้นใต้ และยังให้ความแตกต่างกันระหว่างตัวอักษรเล็กและตัวอักษรใหญ่ในรูปแบบที่เรียกว่า เคสเซนซิทีฟ (Case Sensitive) ด้วยเช่น FILE1, file1 และ File1 นั้นทั้ง 3 ไฟล์นี้จะถือว่าเป็นคนละชื่อกัน

สำหรับการตั้งชื่อไฟล์นั้นควรหลีกเลี่ยงการตั้งชื่อด้วยเครื่องหมายพิเศษต่างๆเช่น ^ ” ‘ , - ? [ ] ( ) ~ ! $ { } < > # @ & / และยิ่งกว่านั้นถ้าไฟล์ใดที่มีชื่อที่ขึ้นต้นด้วย ดอท(“.”) จะกลายเป็นไฟล์ที่ถูกซ่อนไว้ (Hiddden file) ซึ่งจะไม่แสดงรายชื่อไฟล์ออกมาให้เห็นเมื่อดูด้วยวิธีปกติ หากต้องการเรียกไฟล์ที่ถูกซ่อนขึ้นมาดูต้องเพิ่มพารามิเตอร์พิเศษให้กับคำสั่งในการขอดูรายชื่อไฟล์

ตารางที่ 2.2 ระบบชื่อและขนาดของไฟล์ในรูปแบบต่างๆ

ระบบไฟล์	ขนาดของชื่อไฟล์(ตัวอักษร)	ขนาดไฟล์ (เมกะไบต์)
Minix File System	14	64
Extended File System(Ext1)	255	2048
Extended File System(Ext2)	255	2048

2.2.2 ระบบไฟล์ในระบบลินุกซ์

ขนาดของไฟล์ในระบบลินุกซ์ รุ่นแรกๆนั้นไม่สามารถเก็บไฟล์ที่มีขนาดใหญ่เกินกว่า 64 เมกะไบต์ได้ทำให้ผู้ใช้งานที่ต้องการที่จะใช้ไฟล์ขนาดใหญ่บนลินุกซ์ ไม่สามารถทำได้ จึงได้มีการออกแบบและสร้างระบบไฟล์ขึ้นใหม่เรียกว่า Ext (Extended File System) เมื่อเดือนเมษายน ปี 2535 เป็นระบบไฟล์ที่สองที่ระบบปฏิบัติการลินุกซ์ สนับสนุนการทำงานด้วย โดยระบบไฟล์นี้ถูกสร้างโดย "เรมี คาร์ด" (Remy Card) เพื่อให้ระบบสามารถสนับสนุนการใช้งานไฟล์ขนาดใหญ่ โดยระบบไฟล์นี้มีพื้นฐานมาจากระบบไฟล์ใน Minix ทำให้มีข้อจำกัดในระบบไฟล์ของระบบลินุกซ์ลดลง แต่ผลจากการออกแบบส่วนจัดการพื้นที่ว่างที่ยังไม่ดีพอทำให้ระบบจะเกิดพื้นที่ว่างที่ไม่ต่อเนื่องกันได้ง่ายทำให้เกิดปัญหาที่เรียกว่าแฟรกเมนต์เทชั่น (Fragmentation) ส่งผลให้ระบบทำงานช้ากว่าระบบของ Minix ที่เป็นระบบไฟล์ต้นแบบต่อมา ""เรมี คาร์ด" และ "เวย์น่ ดาวิสัน" (Wayne Davison) ได้ทำการออกแบบระบบไฟล์ใหม่เรียกว่าระบบไฟล์แบบ Ext2 (Extended File System 2) ในเดือนมกราคม ปี 2536 หลังจากที่เก็บข้อมูลความต้องการจากผู้ใช้ที่ต้องการระบบไฟล์ที่มีประสิทธิภาพ เป้าหมายในการออกแบบระบบไฟล์นี้จึงมุ่งไปที่ประสิทธิภาพและความถูกต้องของข้อมูล และสามารถรองรับขนาดของพาร์ทิชันที่สูงถึง 4 เทราไบต์ระบบไฟล์นี้มีพื้นฐานมาจากระบบไฟล์ Ext ระบบไฟล์นี้เป็นที่ยอมรับมาจนถึงปัจจุบัน

2.2.3 การจัดแบ่งพาร์ทิชันของระบบไฟล์

การอ้างถึงพาร์ทิชันต่างๆในระบบลินุกซ์ นั้นจะมองไฟล์ที่จดเก็บไว้ในแต่ละพาร์ทิชันต่อเนื่องกันตลอดทุกพาร์ทิชัน โดยเริ่มจากไดเร็กทอรีราก (root directory) ต่อเนื่องไปยังไดเร็กทอรีต่างๆที่ถูกจัดเก็บไว้ในแต่ละพาร์ทิชัน แต่อย่างไรก็ตามระบบปฏิบัติการจะไม่ได้มองพื้นที่ว่างทั้งหมดในแต่ละพาร์ทิชันเป็นพื้นที่ว่างพื้นที่เดียวกันหมด

การจัดการระบบไฟล์ของลินุกซ์ แบบนี้ทำให้การจัดเก็บไฟล์ในแต่ละไดเร็กทอรีนั้นเก็บในพาร์ทิชันใด ดังนั้นไฟล์ที่จัดเก็บจึงต้องเก็บในพาร์ทิชันเดียวกันด้วย ผลจากการจัดการไฟล์ด้วยวิธีการนี้จึงต้องมีการกำหนดจุดเชื่อมต่อ (Mount Point) ให้กับแต่ละพาร์ทิชันต่างๆ เพื่อให้เข้าถึงพาร์ทิชันทั้งหมดได้ ไดเร็กทอรีรากจะเก็บไฟล์โปรแกรมและไฟล์คุณสมบัติที่จำเป็นในการบู๊ตระบบ และไดเร็กทอรีที่ใช้ในการเชื่อมต่อกับพาร์ทิชันอื่นๆ ดังนั้นพาร์ทิชันหลัก (root partition หรือ main partition) จะถูกเชื่อมต่อเข้ากับระบบปฏิบัติการลินุกซ์ เป็นลำดับแรก ถ้าพาร์ทิชันหลักเสียหายจะทำให้ไม่สามารถบู๊ตหรือเริ่มต้นการทำงานของระบบปฏิบัติการได้ ด้วยวิธีการนี้ระบบไฟล์ของระบบ Linux จึงมีเพียงหนึ่งไดเร็กทอรีแต่สามารถทำงานบนระบบที่มีฮาร์ดดิสกี่ตัวก็ได้

การที่ระบบปฏิบัติการลินุกซ์ ทำการแยกย่อยฮาร์ดดิสออกเป็นหลายๆพาร์ทิชันแทนการใช้พาร์ทิชันขนาดใหญ่เพียงหนึ่งพาร์ทิชัน เพราะต้องการให้แต่ละพาร์ทิชันมีการจองขนาดการใช้พื้นที่ใช้สอยไว้ล่วงหน้า เพื่อป้องกันการใช้พื้นที่ว่างจนหมดแล้วทำให้ระบบปฏิบัติการไม่สามารถทำงานต่อไปได้ ยกตัวอย่างเช่น ถ้ามีการติดตั้งโปรแกรมประยุกต์จนเต็มพื้นที่ใช้สอยทั้งหมดจะทำให้ระบบปฏิบัติการไม่สามารถทำงานต่อไปได้

การจัดเก็บไฟล์ในระบบปฏิบัติการลินุกซ์ จะใช้วิธีเก็บในรูปของบล็อก (Block) โดยจะมีขนาดจำนวนเท่าของ 512 ไบต์ ขึ้นอยู่กับการกำหนดขนาดของระบบนั้นๆ ซึ่งการกำหนดขนาดเล็กหรือใหญ่ก็มีข้อดีข้อด้อยแตกต่างกันไป หากบล็อกมีขนาดใหญ่จะทำให้การแลกเปลี่ยนข้อมูลระหว่างดิสก์กับหน่วยความจำทำได้รวดเร็วแต่จะทำให้เกิดการสูญเสียพื้นที่ (Fragmentation) ไปได้ โดยง่ายการจัดเก็บโดยปกติจะเก็บเป็นบล็อกที่ติดต่อกัน แต่ถ้ามีไฟล์อื่นเข้ามาแทรกก็จะสามารถกระโดดข้ามบล็อกนั้นไปได้ เนื่องจากจะมีตารางบอกว่าไฟล์นั้นเก็บข้อมูลไว้ที่บล็อกใดบ้างโดยเก็บไว้ในส่วนของไอโหนด

2.2.4 การจัดการไดเร็กทอรีของระบบไฟล์ลินุกซ์

ลินุกซ์นั้นใช้โครงสร้างของไฟล์และไดเร็กทอรีแบบของ FSSTND (Linux File System Standard) ซึ่งต่อมาได้มีระบบหลายระบบนำรูปแบบการจัดการไฟล์และไดเร็กทอรีแบบนี้ไปใช้และพัฒนาเป็นระบบโครงสร้างใหม่เรียกว่า FHS (Filesystem Hierarchy Standard) แทน โดยมีการจัดเป็นแบบลำดับชั้น โดยมีลักษณะคล้ายกับต้นไม้กลับหัว โดยจุดเริ่มต้นหรือชั้นแรกจะเรียกว่าราก (root) ซึ่งจะเขียนแทนด้วยเครื่องหมาย / ไฟล์แต่ละไฟล์นั้นอาจจะเป็นข้อมูลที่สร้างขึ้น หรือโปรแกรมก็ได้ หรือแม้แต่ใช้เป็นที่เก็บไฟล์เองก็ได้ ซึ่งไฟล์ลักษณะนี้จะจัดว่าเป็นไฟล์ประเภท

ไดเร็กทอรีรูปแบบการจัดเก็บไฟล์แบบลำดับชั้นนี้จะทำให้การจัดเก็บไฟล์เป็นระบบ ทำให้ง่ายต่อการใช้งานและดูแลโครงสร้างหลักของการจัดการไฟล์

2.2.5 ไอโหนด

เคอร์เนลจะไม่สามารถติดต่อหรือรู้จักไฟล์โดยตรงแต่เคอร์เนลจะรับรู้สิ่งต่างๆ เกี่ยวกับไฟล์ที่มีอยู่ได้จากโครงสร้างข้อมูลที่เรียกว่าไอโหนด โดยไอโหนดจะประกอบด้วยข้อมูลต่างๆ เกี่ยวกับไฟล์เป็นจำนวนมาก เช่น การแสดงสิทธิต่างๆของไฟล์

- การบอกถึงชนิดของไฟล์
- แสดงถึงเจ้าของและกลุ่มของเจ้าของไฟล์
- วัน เวลา ที่สร้างไฟล์ เปลี่ยนแปลงไฟล์ หรือแอกเซสไฟล์
- จำนวนลิงค์ที่ไฟล์เชื่อมต่ออยู่ด้วย
- เก็บตำแหน่งที่อยู่ของข้อมูลในไฟล์

ไอโหนดจะถูกสร้างขึ้นมาโดยอัตโนมัติตอนสร้างระบบไฟล์ และถ้าหากไอโหนดนั้นเต็มจะไม่สามารถสร้างไฟล์ขึ้นมาได้เลยถึงแม้ว่าจะยังมีที่ว่างเหลืออยู่

2.3 การกู้ข้อมูล(File recovery)

ในการกู้คืนไฟล์ใช้ในกรณีที่ถูกผู้ไม่ประสงค์ดีลบ หรือทำลายข้อมูลที่อาจใช้เป็นหลักฐาน หรือแม้แต่ในกรณีที่ผู้ใช้ลบหรือทำลายข้อมูลที่อาจใช้เป็นหลักฐานโดยตั้งใจหรือไม่ได้ตั้งใจก็ตาม โดยลักษณะของการทำลายหลักฐานของผู้ไม่ประสงค์ดีมีอยู่สองรูปแบบได้แก่

- ลบ(delete)
- ฟอแมต(Format)

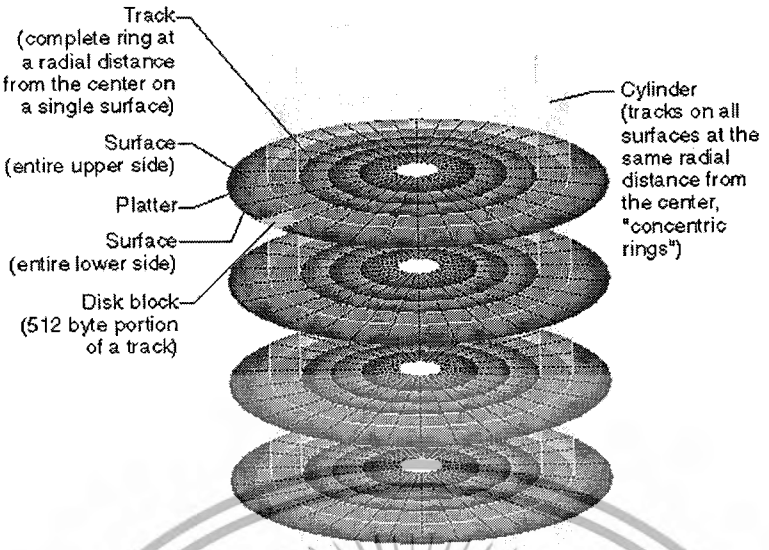
โดยในโครงงานนี้จะเราจะสร้างแอพพลิเคชั่นที่ทำงานร่วมกับระบบไฟล์ซิสเตม FAT32 เพียงเท่านั้น ด้วยเหตุผลอยู่ 3 ประการด้วยกัน

- ไฟล์ซิสเตมแบบ FAT32 เป็นระบบไฟล์ที่ใช้ได้กับทุกระบบปฏิบัติการ
- มีความซับซ้อนน้อยกว่าระบบไฟล์อื่นเหมาะแก่การศึกษาและพัฒนาในระยะเวลาสั้นๆ
- เป็นระบบที่มีการเปิดเผยข้อมูลค่อนข้างมากพอสมควร

2.3.1 โครงสร้างของการเก็บข้อมูลระดับต่ำ

ฮาร์ดดิสก์ ประกอบด้วยแผ่นจานโลหะแข่งที่เรียกว่าแพลตเตอร์(platter) วางซ้อนกันโดยมีช่วงว่างระหว่างแผ่น และเพื่อที่จะจัดเก็บข้อมูลไว้ได้อย่างมีประสิทธิภาพและสะดวกต่อการจัดการ จึงได้แบ่ง โครงสร้างการเก็บข้อมูลระดับต่ำ ให้ประกอบด้วย 3 อย่างได้แก่

- **แทรค(Track)** พื้นผิวบนจานแม่เหล็กบนฮาร์ดดิสก์จะถูกแบ่ง ออกเป็นวงรอบเรียกว่า Track ซึ่งจะอ้างอิงถึงแต่ละ แทรค ได้เพราะแต่ละ แทรคจะมีลำดับเลขกำกับ โดยที่ แทรค ที่ 0 จะอยู่วงนอกสุด ส่วน แทรคที่ 1, 2 .... จะเป็นวงรอบถัดไปด้านในตามลำดับ ดังนั้น Track ที่มีลำดับเลขมากที่สุดจะอยู่วงในสุด โดยวงนอกจะเก็บข้อมูลได้มากกว่าวงใน ซึ่งฮาร์ดดิสก์โดยทั่วไปจะมี แทรค ประมาณ 2000 TPI (Track per inch)
- **เซคเตอร์(Sector)** แต่ละ แทรค จะแบ่งออกเป็นกลุ่มย่อยของข้อมูลตามแนวรัศมีของวงกลม คล้ายการแบ่งขนมเค้ก เรียกว่า เซคเตอร์ ซึ่งแต่ละ เซคเตอร์ โดยปกติจะสามารถเก็บข้อมูลได้ 512 ไบต์ และแต่ละ Sector จะถูกกำกับด้วยตัวเลขอ้างอิง โดยปกติแล้วการที่ แทรค วงนอกมีพื้นที่มากกว่า แทรค วงใน ในขณะที่ เซคเตอร์ ในแต่ละ แทรค นั้นมีจำนวนเท่ากัน ได้ทำให้เกิดการสิ้นเปลืองพื้นที่บนจานแม่เหล็กได้ จึงได้มีวิธีการ Format ข้อมูลแบบใหม่ที่สามารถทำให้จำนวนเซคเตอร์ ของ แทรค วงนอกมีจำนวนมากกว่า แทรค วงในได้ ซึ่งทำให้ฮาร์ดดิสก์สามารถจุข้อมูลได้เพิ่มขึ้น ซึ่งเราจะเรียกเทคนิคนี้ว่า มัลติพัล โซน เรคร์อดดิ่ง (Multiple Zone Recording)
- **ไซลินเดอร์(Cylinder)** เป็นการจัดกลุ่มของ Track หมายเลขเดียวกันของจานแม่เหล็กทุกๆแผ่น ดังนั้น ไซลินเดอ จึงมีลักษณะเป็นทรงกระบอก เนื่องจากฮาร์ดดิสก์จะประกอบด้วยหลายหัวอ่าน จึงสามารถอ้างอิงได้ว่าจะเข้าถึงโดยระบุหมายเลขของ ไซลินเดอ , เฮด(Head) และ Sector ประโยชน์ของการแบ่งข้อมูลเป็นไซลินเดอก็คือเป็นการเข้าถึงข้อมูลได้โดยไม่ต้องเสียเวลาในการเลื่อนหัวอ่านหากข้อมูลอยู่ในไซลินเดอ เดียวกัน

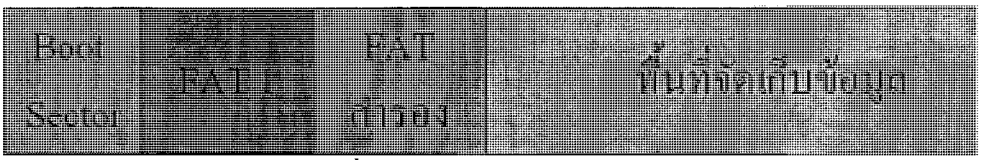


รูปที่ 2.3 แสดงตัวอย่างโครงสร้างของฮาร์ดดิสก์

2.3.2 โครงสร้างระบบไฟล์ FAT

ระบบไฟล์ FAT ถูกพัฒนาโดยไมโครซอฟท์ขึ้นมาพร้อมกับ DOS เวอร์ชันแรก ซึ่งขณะนั้นยังเป็น FAT แบบ 12 บิต ที่ใช้กับ ฟล็อปปี้ดิสก์เท่านั้น ต่อมาจึงได้พัฒนา FAT แบบ 16 บิต เพื่อใช้กับฮาร์ดดิสก์ โดยระหว่างนี้ก็ได้มีการเพิ่มเติมขีดความสามารถของ FAT16 ให้รองรับฮาร์ดดิสก์และโวลุมให้มีขนาดใหญ่ขึ้น แต่ก็ยังถูกจำกัดให้ขนาดของโวลุมหนึ่ง ๆ ได้แค่ 32 เมกะไบต์ จากนั้นใน DOS 4.0 ก็ได้นำระบบ คลัสเตอร์เข้ามาให้งาน ส่งผลให้โวลุมมีขนาดได้สูงสุดประมาณ 2 Giga Byte แต่เมื่อขนาดของฮาร์ดดิสก์ มีขนาดใหญ่ขึ้น ตัวเลข 2 กิกะไบต์ จึงกลายเป็นข้อจำกัดอีกครั้ง และเนื่องจาก FAT16 ได้มาถึงทางตันแล้ว ในที่สุดไมโครซอฟท์ จึงได้ตัดสินใจออกแบบระบบไฟล์ใหม่ที่มีความยืดหยุ่นและรองรับโวลุมได้สูงสุดถึง 2 เทลาไบต์ โดยเปิดตัวพร้อมกับ Windows 95 และใช้ชื่อว่า FAT32 โวลุมที่ใช้ FAT32 จะสามารถเพิ่มความยืดหยุ่นในการใช้งาน โดย FAT32 ประกอบด้วยเอ็นทรีขนาด 32 บิต แต่ใช้จริงได้อย่างมาก 28 บิต โดยสงวนไว้ 4 บิต โดยโครงสร้างข้อมูลของ FAT32 คือ

- พื้นที่สงวน
- FAT(File Allocation Table)
- พื้นที่จัดเก็บข้อมูล



รูปที่ 2.4 แสดงโครงสร้างของ FAT32

2.3.2.1 พื้นที่สงวน

เป็นส่วนที่ใช้เก็บรายละเอียดเบื้องต้นของระบบไฟล์ซิสเตม โดยปกติใน FAT32 จะมีเพียงแค่ 3 เซคเตอร์เท่านั้น โดยเอนทรีต่าง ๆของพื้นที่สงวนที่สำคัญ ๆ มีตามตารางที่ 2.3 ดังนี้

ตารางที่ 2.3 แสดง บูตเรกคอร์ด(boot record) ของ FAT32

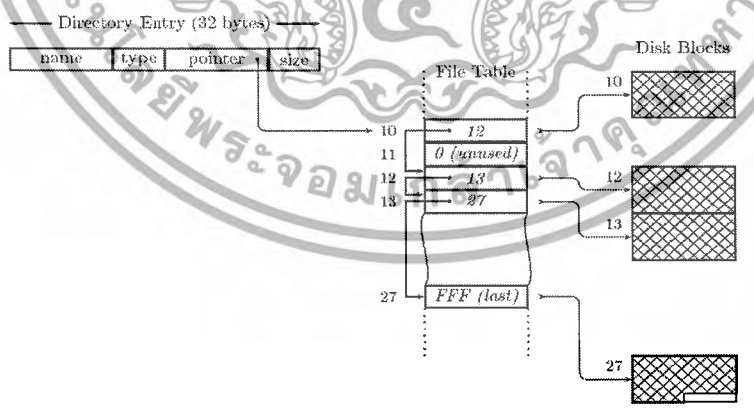
ออฟเซต	รายละเอียด	ขนาด
00h	เป็นคำสั่งให้กระโดดไปยังชุดคำสั่งที่ใช้บู๊ตระบบ	3 ไบต์
03h	ชื่อผู้ผลิตและระบบปฏิบัติการ	8 ไบต์
0Bh	จำนวนไบต์ต่อเซคเตอร์	1 เวิร์ด
0Dh	จำนวนเซคเตอร์ต่อคลัสเตอร์	1 ไบต์
0Eh	จำนวนเซคเตอร์ของพื้นที่สงวน	1 เวิร์ด
10h	จำนวนชุดของ FAT	1 ไบต์
11h	จำนวนเอ็นทรีของรูทไดเร็คทอรี	1 เวิร์ด
13h	จำนวนเซคเตอร์ในโวลุม(เป็น 0 ถ้ามีมากกว่า 65535)	1 เวิร์ด
15h	รายละเอียดของสื่อ(Media descriptor)	1 ไบต์
16h	จำนวนเซคเตอร์ของ FAT	1 เวิร์ด
18h	จำนวนเซคเตอร์ต่อแทรคของฮาร์ดดิสก์	1 เวิร์ด
1Ah	จำนวนหัวอ่าน/ เขียนของฮาร์ดดิสก์	1 เวิร์ด
1Ch	จำนวนเซคเตอร์ที่ซ่อนไว้	1 ดับเบิล เวิร์ด
20h	จำนวนเซคเตอร์ในโวลุม	1 ดับเบิล เวิร์ด
24h	จำนวนเซคเตอร์ของFAT	1 ดับเบิล เวิร์ด

ตารางที่ 2.3 แสดง บูตเรกคอร์ด(boot record) ของ FAT32(ต่อ)

28h	แฟลก	1 เวิร์ด
2Ah	เวอร์ชั่น	1 เวิร์ด
2Ch	หมายเลขคลัสเตอร์แรกของรูทไดเรกทอรี	1 ดับเบิล เวิร์ด
43h	ซีเรียว นัมเบอร์	1 ดับเบิล เวิร์ด
52h	ข้อความบอกชนิดของ FAT	8 ไบต์

2.3.2.2 FAT(File Allocation Table)

FAT ประกอบด้วยเอ็นทรีขนาดคงที่ต่อเนื่องกันไป ขนาดของเอ็นทรีที่ว่านี้คือ 12 บิต สำหรับระบบไฟล์ FAT12 , 16 บิตสำหรับ FAT16 และแน่นอน 32 บิต สำหรับ FAT32 ระบบปฏิบัติการจะมองเอ็นทรีของ FAT เหล่านี้ว่าเป็นหมายเลข 0,1,2,... ไปเรื่อย ๆ แต่ละเอ็นทรีนั้นจะสัมพันธ์แบบหนึ่งต่อหนึ่งกับพื้นที่ใดส่วนหนึ่งของพื้นที่ไฟล์ด้วยลำดับที่ตรงไปตรงมา ซึ่งใน DOS ยุคแรก ๆ จะอ้างกับเซคเตอร์ แต่เมื่อความต้องการหน่วยความจำในฮาร์ดดิสก์เพิ่มสูงขึ้น ไมโครซอฟท์จึงได้คิด การสร้างความสัมพันธ์ ระหว่างเอ็นทรีใน FAT กับเซคเตอร์ โดยแทนที่จะเป็นเซคเตอร์เดียว ก็อ้างกับหลาย ๆ เซคเตอร์ โดยใช้ชื่อว่า คลัสเตอร์(cluster) โดยคลัสเตอร์อาจประกอบด้วยเซคเตอร์จำนวน 1,2,4,8,16,32,64 หรือ 128 เซคเตอร์ ขึ้นอยู่กับความจุของโวลุม



รูปที่ 2.5 แสดงกลไกการเชื่อมโยงระหว่างไฟล์หนึ่งกับFATชนิด FAT32

นอกจากการชี้ไปยังคลัสเตอร์ถัดไปแล้ว เอ็นทรีใน FAT ก็อาจเก็บค่าพิเศษที่สื่อความหมายอื่น ๆ ดังนี้

ตารางที่ 2.4 แสดงความหมายของค่าต่าง ๆ ในเอ็นทรีของ FAT32

FAT32	ความหมาย
0x?0000000	คลัสเตอร์ว่างใช้งานได้
0x?0000001	ถูกสงวนไว้ ห้ามใช้
0x?0000002 - 0x?FFFFFEF	เป็นค่าที่ชี้ไปยังคลัสเตอร์ถัดไป
0x?FFFFFF0 - 0x?FFFFFF6	ถูกสงวนไว้ ห้ามใช้
0x?FFFFFF7	คลัสเตอร์ใช้การไม่ได้
0x?FFFFFF8 - 0x?FFFFFFF	คลัสเตอร์สุดท้ายของไฟล์

2.3.2.3 รูทไดเร็กทอรี(Root Directory)

เป็นส่วนที่อยู่ถัดจาก FAT ใน FAT12/16 ได้กำหนดให้รูทไดเร็กทอรีอยู่ในตำแหน่งที่คงที่ คือถัดจาก FAT แต่ใน FAT32 นั้น ถือว่ารูทไดเร็กทอรีเป็นเพียงไฟล์ธรรมดาไฟล์หนึ่ง จะต่างจากไฟล์อื่นก็ตรงที่หมายเลขคลัสเตอร์แรกของรูทไดเร็กทอรีถูกเก็บไว้ในบูตเรกคอร์ดแทนที่จะเป็นไดเร็กทอรีเอ็นทรี เหตุผลที่รูทไดเร็กทอรีไม่ถูกกำหนดตำแหน่งและขนาดตายตัวก็เพราะว่า ต้องการให้สามารถขยายได้ตามขนาดของโวลุม

รูทไดเร็กทอรีประกอบด้วยเอ็นทรีต่าง ๆ ที่มีขนาดตายตัวเช่นเดียวกับโครงสร้าง FAT เอ็นทรีเหล่านี้ทำหน้าที่เก็บรายละเอียดเบื้องต้นของไฟล์และไดเร็กทอรี โดย FAT32มีโครงสร้างรายละเอียดดังนี้

ตารางที่ 2.5 แสดงค่าฟิลด์ต่าง ๆ ในรูทไดเร็กทอรีของ FAT32

ออฟเซต	รายละเอียด	ขนาด
00h	ชื่อไฟล์	8 ไบต์
08h	นามสกุลของไฟล์	3 ไบต์
0Bh	แอตทริบิวต์ของไฟล์	1 ไบต์
0Ch	สงวนไว้	1 ไบต์

ตารางที่ 2.5 แสดงค่าฟิลด์ต่าง ๆ ในรูทไดเร็กทอรีของ FAT32(ต่อ)

0Dh	เวลาที่สร้างไฟล์หน่วยเป็น millisecond	1 ไบต์
0Eh	เวลาที่สร้างไฟล์	1 เวิร์ด
10h	วัน/เดือน/ปี ที่ไฟล์ถูกสร้างขึ้น	1 เวิร์ด
12h	วัน/เดือน/ปี ที่ไฟล์ถูกเข้าถึงครั้งล่าสุด	1 เวิร์ด
14h	หมายเลขคลัสเตอร์แรกของไฟล์ไบต์สูง	1 เวิร์ด
16h	เวลาที่ไฟล์ถูกแก้ไขครั้งล่าสุด	1 เวิร์ด
18h	วัน/เดือน/ปี ที่ไฟล์ถูกแก้ไขครั้งล่าสุด	1 เวิร์ด
1Ah	หมายเลขคลัสเตอร์แรกของไฟล์ไบต์ต่ำ	1 เวิร์ด
1Ch	ขนาดของไฟล์	1 ดับเบิลเวิร์ด

2.3.2.4 พื้นที่จัดเก็บข้อมูล (File Area)

เป็นส่วนที่ใช้จัดเก็บเนื้อหาของไฟล์และไดเร็คทอรีย่อยต่าง ๆ ไว้ โดยหน่วยที่เราให้ความสนใจก็คือ คลัสเตอร์

2.3.3 ชื่อไฟล์แบบยาว (Long File name)

ในการตั้งชื่อตัวอักษร ตัวอักษรจำนวน 8 ตัวไม่สามารถสื่อสารอะไรได้มากนัก ไมโครซอฟท์ซึ่งได้เลงเห็นปัญหานี้จึงได้คิดวิธีการที่สามารถตั้งชื่อไฟล์ได้มากถึง 255 ตัวอักษร ซึ่งวิธีการที่ผู้พัฒนาได้ใช้นั้นไม่จำเป็นต้องปรับปรุงโครงสร้างของ FAT เลย เพียงแต่เซตแอตทริบิวต์อ่านเท่านั้น(Read-only),ซ่อน(Hidden),ระบบ(System),โวลุม ลาเบล(Volume Label) ให้กับไดเร็คทอรีเอ็นทรีเป็น 1 ทั้งหมด และ หมายเลขคลัสเตอร์แรกที่กำหนดไว้ในไดเร็กทอรีเอ็นทรีเซตให้เป็น 0 จากวิธีนี้ทำให้สามารถเพิ่มชื่อไฟล์แบบยาวไปในระบบ โดยไม่ส่งผลกระทบต่อซอฟแวร์เก่าเลย โดยโครงสร้างของไดเร็กทอรีเอ็นทรีที่ใช้เก็บไฟล์แบบยาวมีดังนี้

ตารางที่ 2.6 แสดงโครงสร้างของไดเร็กทอรีที่ใช้เก็บชื่อไฟล์แบบยาว

ออฟเซต	รายละเอียด	ขนาด
00h	หมายเลขลำดับของเอ็นทรี	1 ไบต์
01h	ชื่อไฟล์	10 ไบต์
0Bh	แอตทริบิวต์ มีค่าเป็น 0Fh เสมอ	1 ไบต์
0Ch	ชนิดเอ็นทรี มีค่าเป็น 00h เสมอ	1 ไบต์
0Dh	ค่า Checksum	1 ไบต์
0Eh	ชื่อไฟล์(ต่อ)	1 ไบต์
1Ah	หมายเลขคลัสเตอร์แรกของไฟล์ มีค่า 0000h เสมอ	1 เวิร์ด
1Ch	ชื่อไฟล์(ต่อ)	4 ไบต์

เมื่อเทียบกับฟิลด์ในรูทไดเร็กทอรีปกติ ฟิลด์ที่ยังคงเดิมทั้งขนาดและตำแหน่งก็คือ แอตทริบิว(ออฟเซตที่ 0BH) และ หมายเลขของคลัสเตอร์แรก (ออฟเซตที่ 1Ah) แต่มันไม่ได้สื่อความหมายดั้งเดิมของมัน แต่มีไว้ใช้หลอกซอฟต์แวร์เก่า ๆ ให้เพิกเฉยต่อเอ็นทรีแบบยาว

เอ็นทรีแบบสั้นและเอ็นทรีแบบยาวของไฟล์หนึ่ง ๆ จะอยู่ติดกันเสมอ โดยเริ่มต้นจากเอ็นทรีแบบบยาวก่อนแล้วจึงตบท้ายด้วยเอ็นทรีแบบสั้น

2.4 การกู้รหัสผ่าน(Password Recovery)

การถอดรหัสผ่านนั้น เป็นเรื่องที่หลายคนให้ความสนใจคงเป็นเพราะลักษณะนิสัยของมนุษย์เราที่ว่าสิ่งใดที่ไม่ต้องการให้รู้ยิ่งอยากรู้ สิ่งใดที่ไม่ต้องการให้เห็นยิ่งอยากเห็น ปัจจุบันนี้ความปลอดภัยของข้อมูลเป็นสิ่งที่ทุกคนมีความกังวลกันมากขึ้น เพราะว่ามีเครื่องมือที่ใช้ในการถอดรหัสผ่าน (Password Cracking) ที่ไม่ว่าจะเป็นใครก็สามารถที่จะหามาใช้ได้ไม่ยากในอินเตอร์เน็ต มีทั้งแบบที่แจกให้ฟรี และแบบที่เสียเงินซื้อ เครื่องมือแต่ละตัวก็มีความสามารถที่แตกต่างกันไปตามแต่ที่ผู้พัฒนาคิดค้นขึ้น โดยเครื่องมือเหล่านี้ผู้พัฒนามีวัตถุประสงค์การใช้งานเพื่อช่วยกู้รหัสผ่าน (Password Recovery) ในยามที่เกิดเหตุการณ์คับขัน เช่นคุณลืมรหัสผ่านต่างๆที่ตนเองได้กำหนดไว้ ไฟล์ที่มีการกำหนดสิทธิการใช้งานโดยใช้รหัสผ่าน (Password) ย่อมเป็นไฟล์

หรือข้อมูลที่มีความสำคัญ ดังนั้นเครื่องมือที่ใช้ในการกู้รหัสผ่าน (Password Recovery Tool) จึงเป็นเครื่องมือที่มีประโยชน์อย่างมาก

เนื่องจากเราได้ศึกษาเกี่ยวกับเรื่อง Computer Forensic ซึ่งเป็นเรื่องของการค้นหา และพิสูจน์หลักฐานดิจิตอล เพื่อใช้เป็นหลักฐานในการระบุผู้กระทำผิด จนถึงเป็นหลักฐานในชั้นศาลได้ การกู้รหัสผ่าน (Password Recovery) จึงได้ถูกนำมาใช้เพื่อที่จะกู้รหัสผ่านของไฟล์ หรือข้อมูลที่เราได้เก็บมาเป็นหลักฐาน เพราะว่าไฟล์นั้นอาจถูกผู้กระทำผิดทำการเข้ารหัสไว้ หรือตั้งรหัสผ่านไว้เพื่อที่จะได้หลักฐานในการจับกุมจึงต้องถอดรหัสไฟล์นั้นๆมาให้ได้

2.4.1 รหัสผ่าน(Password)

พาสเวิร์ด(Password) คือ อตัวอักษรหรือคำที่ใช้รับรองว่าเป็นผู้ใช้จริง ระบบคอมพิวเตอร์จะถามชื่อผู้ใช้ (or login name) และ พาสเวิร์ด (or pass phrase) ก่อนที่จะยอมรับเขาทั้งหลายเข้าใช้ระบบถ้าเป็นบุคคลที่รู้ชื่อผู้ใช้ และพาสเวิร์ด ระบบคอมพิวเตอร์จะไว้วางใจให้เขาทั้งหลายนั้นเข้าใช้ระบบของคุณ และอนุญาตให้เข้าถึงข้อมูลได้

2.4.2 กำหนดรหัสผ่าน (Password Construction)

การถอดรหัสนั้นมีอุปกรณ์มากมายที่เขาใช้จัดการตัดจำนวนเวลาในการถอดรหัสพาสเวิร์ดของคุณ การใช้รหัสผ่านที่มีความปลอดภัยจะช่วยรับประกันได้ว่าการถอดรหัสต้องใช้เวลามาก และมีปัญหาในการคาดเดา อย่างไรก็ตามไม่มีรหัสผ่านไหนที่มีความปลอดภัยที่สุด แต่ถ้าทำให้ การถอดรหัส ใช้เวลานานนากรถอดรหัสกว่าที่เขาจะได้รหัสนั้นไปรหัสนั้นมันอาจจะให้การไม่ได้แล้วก็เป็นได้คุณก็จะประสบความสำเร็จในการขัดขวางการโจมตีของ Cracker ได้

Insecure Methods

- รหัสผ่านไม่ควรเป็นข้อมูลส่วนตัวที่เกี่ยวกับตนเองหรือครอบครัว เพราะการถอดรหัส จะใช้ข้อมูลส่วนตัวที่เกี่ยวกับตัวคุณหรือครอบครัวในการถอดรหัสเป็นอันดับแรกของการเริ่มต้น รหัสผ่านแบบนี้จึงมีความปลอดภัยน้อยที่สุด ยกตัวอย่างเช่น ชื่อตัวคุณเอง , สถานที่เกิด , ชื่อเล่น , ชื่อครอบครัว , ชื่อสัตว์เลี้ยง , ที่อยู่ , ชื่อพ่อแม่
- รหัสผ่านไม่ควรเป็นคำที่มาจาก ดิกชันนารี(dictionary) หรือหนังสือ เพราะว่า การถอดรหัส จะใช้วิธีการโจมตีโดยใช้คำจาก ดิกชันนารี(dictionary) มาเทียบทำให้ง่ายต่อการถอดรหัส ยกตัวอย่างเช่น dragon , secret , love , cheese , god

- รหัสผ่านไม่ควรเป็นคำที่เกิดจากคำที่เปลี่ยนแปลงไปจากเดิม ถึงแม้ว่ารหัสผ่านนี้ไม่มีคำใดที่ปรากฏในหนังสือ หรือ ดิกชันนารี (dictionary) ก็ตาม รหัสผ่านนี้มีความปลอดภัยมากกว่า 2 ตัวอย่างที่ผ่านมา แต่ก็ไม่ปลอดภัยมากนัก ตัวอย่างเช่น l0ve , s3cr3t , dr@gon
- รหัสผ่านไม่ควรเป็นคำนามหรือคำธรรมดา 2 คำรวมกันหรือเชื่อมเข้าด้วยกัน รหัสผ่านแบบนี้มีความปลอดภัยเหนือกว่ารหัสผ่านที่กล่าวมา แต่ยังคงไม่เพียงพอสำหรับการรักษาความปลอดภัย ตัวอย่างเช่น whatfor , bigpig , devineright , ilove , catspajamas

Secure Methods

- เปลี่ยนรหัสผ่านของคุณเสมอๆทันทีที่คุณรู้สึกว่ารหัสผ่านของคุณนั้นเป็นอันตราย คุณควรรีบเปลี่ยนมันทันที
- ไม่เขียนรหัส หรือ รหัสผ่านของคุณลงบนกระดาษหรือสิ่งใดที่คนอื่นสามารถหาได้ง่าย ถ้าคุณจำเป็นที่จะต้องเขียนหรือจด password ของคุณจริงๆต้องรับประกันได้ว่ามันจะอยู่ในที่ๆมีแต่คุณคนเดียวเท่านั้นที่รู้ ซ่อนมันไว้ในที่ๆคุณรู้สึกว่าไม่น่าจะมีใครหาพบ
- การตั้งรหัสผ่านเป็นสิ่งที่สำคัญมากดังนั้นคุณควรเปลี่ยนรหัสผ่าน ของคุณอย่างน้อยทุกๆ 6 เดือน
- กำหนดรหัสผ่านโดยใช้การผสมกันระหว่าง ตัวอักษร , ตัวเลข และตัวอักษรพิเศษอยู่ในรหัสผ่านเดียวจะทำให้ยากต่อการถอดรหัส แต่อย่างไรก็ตามในบางระบบยังไม่ยอมรับให้ใช้อักขระพิเศษ
- กำหนดรหัสผ่านโดยใช้ระบบPass phrase ในระบบ Pass phrase อนุญาตให้ผู้ใช้พิมพ์รหัสผ่านที่ยาวได้ ซึ่งแต่ละตัวอักขระต้องตรงกันกับของระบบจึงจะเข้าระบบได้ผู้ใช้สามารถใช้วลีหรือกลุ่มคำที่สามารถจำได้ ซึ่งทำให้โอกาสในการเดารหัสผ่านได้มีน้อยลง

เราได้พูดถึงการป้องกันข้อมูลต่างๆ โดยใช้รหัสผ่านมาแล้ว รหัสผ่านถูกนำมาใช้เพื่อความเป็นส่วนตัวในชีวิตประจำวัน ในคอมพิวเตอร์มีหลายโปรแกรมที่มีการใช้ระบบกำหนดรหัสผ่านเพื่อสร้างความปลอดภัย เช่น WinZIP , WinRAR , Microsoft Office ไม่ว่าคุณจะใช้โปรแกรมใดๆ ในการจัดทำข้อมูลที่คิดว่ามันน่าจะมีความสำคัญ และต้องการให้เป็นความลับหรือส่วนตัวคงหนีไม่พ้นการกำหนดรหัสผ่าน เพื่อกำหนดสิทธิการเข้าถึงเพื่อป้องกันข้อมูล แต่รหัสผ่านที่กำหนดไว้อาจ

ไม่ทำให้ข้อมูลของคุณปลอดภัยมากนัก เพราะว่าการถอดรหัสผ่านเป็นเรื่องไม่ยุ่งยากเลยในปัจจุบัน วิธีการในการถอดรหัสผ่านนี้เราเรียกกันว่า Password Cracking

การถอดพาสเวิร์ด จะพยายามค้นหาคำในดิกชันนารี ซึ่งดิกชันนารีจะขึ้นอยู่กับประสบการณ์ของ hacker และความรู้เกี่ยวกับระบบที่จะโจมตี ดิกชันนารี(dictionary) จะประกอบด้วยชื่อแรก(first name)ทั่ว ๆ ไป ตัวอักษร,ชื่อเรื่อง,สถานที่ จากนวนิยาย โทรทัศน์และภาพยนตร์,การ์ตูนและเกมส์คอมพิวเตอร์, คำที่เกี่ยวกับกีฬา คำที่เกี่ยวกับอุตสาหกรรมที่คอมพิวเตอร์นั้นใช้อยู่ คำทั้งหมดที่กล่าวมาข้างต้นจะถูกนำมาจัดเรียงด้วยลักษณะดังต่อไปนี้

- ตัวอักษรตัวใหญ่และตัวอักษรตัวเล็กหลายแบบ
- สลับที่การสะกดคำ
- เปลี่ยนตัวเลข 0,1,2 และ 5 สำหรับตัวอักษร o,i,z และz ในคำนั้น
- เพิ่มตัวเลขหนึ่งตัวเข้าไปในคำนั้น
- จับคู่คำสองคำเข้าด้วยกัน คั่นระหว่างคำสองคำด้วยอักขระพิเศษ

เนื่องจากรหัสผ่านเป็นวิธีการป้องกันหลักจากบุคคลภายนอก มีการศึกษาจำนวนมากที่เกี่ยวกับเรื่องดังกล่าว จากการศึกษาแสดงให้เห็นว่าระหว่าง 25-30 เปอร์เซ็นต์ของรหัสผ่านจะถูก crack โดยวิธีนี้

จากการศึกษาที่ทำเสร็จในปี คศ.1989 โดย Dan Klein ที่มหาวิทยาลัย Carnegie-Mellon ใช้วิธีการเหล่านี้ เพื่อพยายามถอดรหัสผ่านจริงจำนวน 13,797 account จากหลายแหล่งด้วยกัน ดิกชันนารี ทั้งหมด ประกอบด้วยคำ 62,727 คำ ด้วยวิธีการนี้สามารถเดารหัสผ่านได้ 3,340 รหัส จากตารางนี้ประกอบด้วยข้อมูล ที่ถูกเปิดเผยจากการศึกษาเกี่ยวกับแหล่งของรหัสผ่านที่ถูกเลือกโดยผู้ใช้

ตารางที่ 2.7 แสดงความสามารถในการแกะรหัสผ่าน

แหล่งที่มาของรหัสผ่าน	จำนวนที่ค้น	จำนวนที่ crack ได้	เปอร์เซ็นต์ที่ crack ได้	อัตราประสิทธิภาพ
user/account name	130	368	2.70%	2.830
การจัดเรียงอักขระ	866	22	0.20%	0.025
ตัวเลข	427	9	0.10%	0.021
คำที่เกี่ยวกับจีน	392	56	0.40%	0.143
ชื่อสถานที่	628	82	0.60%	0.131
ชื่อทั่ว ๆ ไป	2239	548	4.00%	0.245
ชื่อผู้หญิง	4280	161	1.20%	0.038
ชื่อผู้ชาย	2866	140	1.00%	0.049
ชื่อแปลก ๆ	4955	130	0.90%	0.026
ตำนานและนิทานปรัมปรา	1246	66	0.50%	0.053
คำที่เกี่ยวกับเชคสเปียร์	473	11	0.10%	0.023
คำที่เกี่ยวกับกีฬา	238	32	0.20%	0.134
นิยายวิทยาศาสตร์	691	59	0.40%	0.085
ภาพยนต์และนักแสดง	99	12	0.10%	0.121
การ์ตูน	92	9	0.10%	0.098
บุคคลที่มีชื่อเสียง	290	55	0.40%	0.190
กลุ่มคำ	933	253	1.80%	0.271
ชื่อเล่น	33	9	0.10%	0.273
ชีววิทยา	58	1	0.00%	0.017

ตารางที่ 2.7 แสดงความสามารถในการแกะรหัสผ่าน(ต่อ)

unix dictionary	19683	1027	7.40%	0.052
เรื่องคอมพิวเตอร์	9018	132	1.00%	0.015
ระบบช่วยความจำ	14	2	0.00%	0.143
King James Bible	7525	83	0.60%	0.011
คำเบ็ดเตล็ด	3212	54	0.40%	0.017
ภาษายิว	56	0	0.00%	0.000
ดาวเคราะห์น้อย	2407	19	0.10%	0.007
รวมทั้งหมด	**62851**	**3340**	**24.30%**	**0.053**

2.4.3 การถอดรหัสผ่าน

การถอดรหัสผ่านมีด้วยกันหลายวิธี เช่น Brute-Force Attack , Dictionary Attack , Plain-Text Attack , Mask

2.4.3.1 Dictionary Attack

Dictionary Attack เป็นวิธีการถอดรหัสชนิดหนึ่งที่นิยมใช้กันมาก โดยปกติทั่วไปแล้ว Dictionary Attack มีความสามารถมากกว่า Brute-Force Attack เพราะว่าผู้ใช้มักตั้งรหัสผ่านแบบง่ายๆกันเป็นส่วนมาก โดยทั่วไประบบแทบจะไม่มีความสำเร็จในการต่อต้านการโจมตีแบบ Dictionary Attack ดังนั้นผู้ใช้จึงต้องใช้ Passphrase ในการตั้งรหัสผ่าน

การโจมตีแบบ Dictionary Attack นั้นจะเป็นการเทียบคำจากรายชื่อในเวิร์ด ไปทีละคำดังนั้นผู้ที่ตั้งรหัสผ่านแบบง่ายๆจึงมีสิทธิที่จะโดนถอดรหัสได้ง่ายเช่นกัน การปรับปรุงแก้ไข Dictionary Attack มีด้วยกัน 2 ทางคือ

- วิธีแรกเพื่อที่จะปรับปรุงแก้ไป Dictionary Attack ให้สมบูรณ์ ผู้ใช้จะต้องมีไฟล์ดิกชันนารี ที่ ใหญ่ มีคำศัพท์มาก มีศัพท์ทางเทคนิค คำศัพท์เฉพาะทาง หรือคำศัพท์ที่เป็นภาษาอื่นๆเพื่อที่จะเพิ่มความสำเร็จในการโจมตี และได้รหัสผ่านที่ถูกต้อง

- วิธีที่สอง ทำการเปลี่ยนแปลงตัวอักษรในดิกชันนารี อย่างเช่น ปกติคำมีว่า password เราก็กลับคำจากหลังไปหน้าเป็น drowssap จะใช้คำอื่นหรือตัวอักษรอื่นหรือตัวเลขมาแทนที่ตัวอักษรเดิม เช่น p4ssw0rd หรือใช้ตัวอักษรใหญ่มาแทนที่ตัวอักษาเล็กที่มีอยู่ เช่น PassWord

ดังนั้นการใช้ Dictionary Attack นั้นจะประสบความสำเร็จขึ้นอยู่กับรายชื่อในดิกชันนารี เพราะถ้ามีคำอยู่มาก และ หลากหลายเท่าไหร่การโจมตีแบบนี้ก็จะประสบความสำเร็จมากเท่านั้น

2.4.3.2 Brute-Force Attack

กระบวนการนี้เป็นการใช้หลักการสร้างรหัสผ่านจากอักขระที่กำหนดขึ้นมาให้เป็นรหัสตั้งต้นซึ่งอาจจะเป็น 0-9 , a-z , A-Z หรืออักขระพิเศษอื่นๆเช่น ~!@#$%^&*()_+ZXM<>?/{}[]:;"',. หรือตัวอักษรใดๆก็ได้ จากนั้นจึงทำการรวมอักขระเหล่านั้นเป็นรหัสตัวใหม่โดยทำการผสมกันตั้งแต่หนึ่งหลักจนมีขนาดหลายๆหลักก็จะกลายเป็นรหัสผ่านได้ ซึ่งวิธีการนี้ถ้าผู้กำหนดรหัสผ่านเป็นผู้ทำการทดสอบเองหรือหากเป็นรหัสผ่านที่มีรูปแบบคล้ายๆกันหรือเรียงกันก็จะสามารถกู้ได้อย่างรวดเร็วกว่าดิกชันนารี ในบางครั้งแต่ก็เป็นหลักการที่คาดคะเนได้ยากเนื่องจากประสิทธิภาพที่ได้ก็ขึ้นอยู่กับความเร็วของการทำงานเมื่อเทียบกับหน่วยเวลาแล้วว่าสามารถทำได้กี่รหัส ซึ่งถ้ามากก็จะเป็นผลดีเรื่องระยะเวลาที่จะใช้งาน แต่ถือว่าเป็นวิธีการที่อาศัยความพยายามเป็นอย่างมากเนื่องจากตัวรหัสจะถูกสร้างออกมาเรื่อยๆเพื่อนำไปใช้โดยที่โปรแกรมหรือผู้ใช้งานเองก็ไม่ทราบเลยว่าจะสำเร็จเมื่อใดทำได้เพียงแต่คำนวณคร่าวๆจากความเร็วของตัวประมวณผลและจำนวณอักระที่ใช้ในการสร้างเท่านั้น แต่รับประกันไม่ได้ว่าจะพบหรือไม่เนื่องจากรหัสอักขระที่ระบุอาจจะมีหรือไม่มีก็ได้

ถ้าการโจมตีแบบ Dictionary Attack ไม่ได้ผล หรือกู้รหัสออกมาไม่ได้ เราอาจจะใช้การโจมตีแบบ Brute-Force Attack แทนถึงแม้ว่าการโจมตีแบบ Dictionary Attack จะมีความรวดเร็วและมีประสิทธิภาพมากกว่า แต่การโจมตีแบบ Brute-Force Attack มีความแน่นอนว่าได้รหัสผ่านที่ถูกต้องมากกว่าแต่โดยมากจะใช้เวลาที่มากกว่าเช่นเดียวกัน

2.5 การสำเนาและกู้คืนข้อมูล (Clone & Restore)

ในกรณีของการสำเนาข้อมูลนั้นใช้หลักการอ่านข้อมูลทุกๆ บิตที่อยู่ในอุปกรณ์ที่ได้ทำการเลือกไว้โดยไม่สนใจว่าในอุปกรณ์นั้นจะมีไฟล์อะไรอยู่ จากนั้นจึงทำการเขียนข้อมูลที่ได้ออกเป็นไฟล์ลงอุปกรณ์ปลายทางที่ทำการเลือกไว้เป็นตัวเก็บข้อมูล ซึ่งในระหว่างขั้นตอนของการสำเนาข้อมูลนั้นหากขนาดของไฟล์ที่ทำการทดลองเกินจากที่กำหนดก็จะทำการปิดไฟล์ที่กำลังทำงานอยู่ จากนั้นจึงทำการสร้างไฟล์ใหม่ขึ้นมาและอ่านเขียนข้อมูลเช่นนี้ไปเรื่อยๆ จนกระทั่งถึงบิตสุดท้าย ก็จะเป็นการอันเสร็จสิ้นกระบวนการในส่วนของการสำเนาข้อมูลจากสื่อต้นฉบับไปสู่สื่อภายนอก

ในการสำเนาข้อมูลนั้นจะได้ไฟล์ที่ทำการบันทึกไว้ เมื่อต้องการที่จะนำไฟล์ที่บันทึกข้อมูลมาทำการตรวจสอบดูก็ต้องทำการ เขียนไฟล์ข้อมูลที่มีอยู่ลงอุปกรณ์ปลายทาง(Restore) อีกครั้งหนึ่ง โดยในกระบวนการนี้เป็นเหมือนการทำงานย้อนกลับจากในส่วนของการสำเนาข้อมูล โดยกระบวนการเริ่มด้วยการเปิดอ่านไฟล์ข้อมูลที่บันทึกไว้และทำการอ่านและเขียนข้อมูลลงอุปกรณ์ปลายทางไปเรื่อยๆจนหมดไฟล์

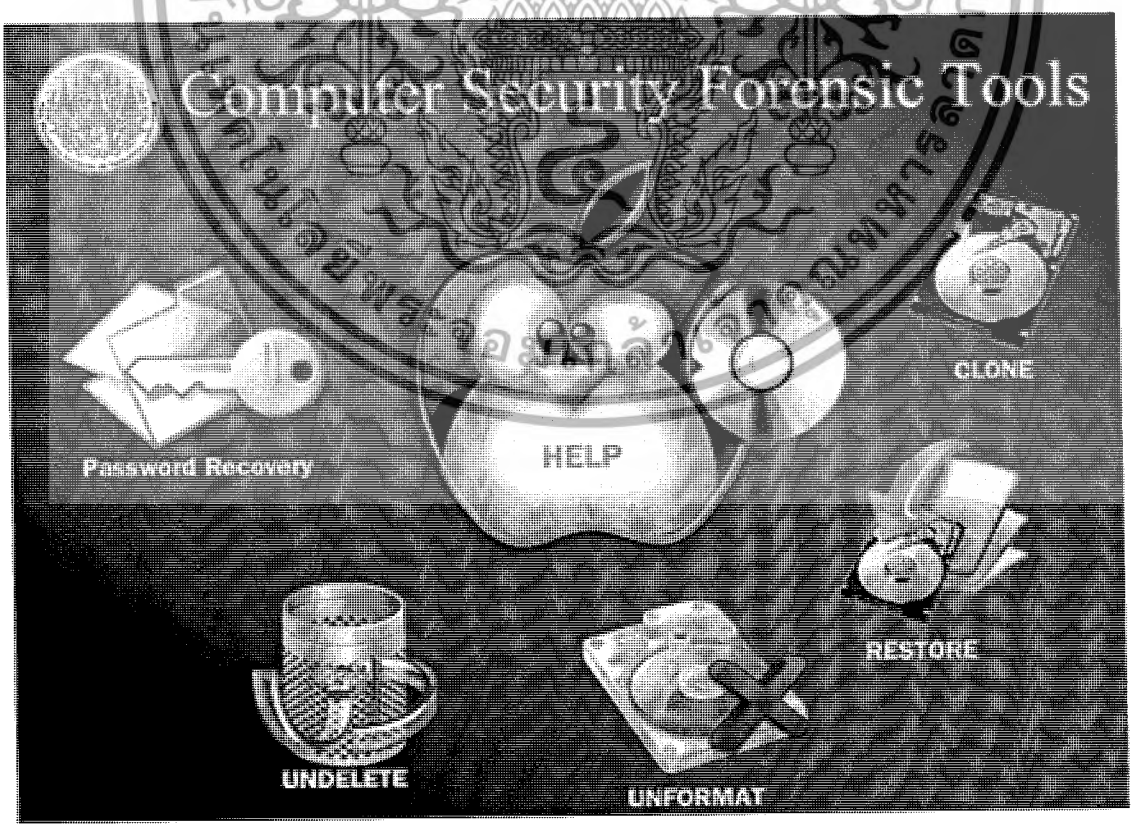
บทที่ 3

การออกแบบและพัฒนา

3.1 การออกแบบ

ในการออกแบบเครื่องมือเก็บหลักฐานในโครงงานนี้ เราจะเน้นไปในทางออฟไลน์ โดยจะมี Live CD เพื่อทำงานเป็นระบบปฏิบัติการบนแผ่นซีดี และมีโปรแกรมโคลน เพื่อที่จะสามารถเข้าไปยังเครื่องที่ถูกกระทำและสามารถคัดลอกฮาร์ดดิสก์ทั้งหมดกลับมาเพื่อนำมาประมวลผลต่อ อีกทั้งยังมีโปรแกรมกู้ข้อมูลเพื่อใช้ในกรณีเครื่องที่ถูกผู้ไม่ประสงค์ดีกระทำการลบหรือทำลายหลักฐานเราก็สามารถกู้ข้อมูลกลับมาได้ และในส่วนสุดท้าย ก็เป็นโปรแกรมถอดพาสเวิร์ดไฟล์ Zip/Rar/7z ซึ่งโปรแกรมทั้งหมดจะสามารถรันอยู่บนแผ่น LiveCD ได้ โดยในการพัฒนาแต่ละโปรแกรมเราจะทำเป็นสองโหมดด้วยกันคือแบบ คอมมานไลน์(command line) และแบบ กราฟฟิคโหมด(GUI) จึงทำให้ผู้ใช้สามารถทำได้ทั้งแบบ คอมมานไลน์ และ แบบกราฟฟิกโหมด

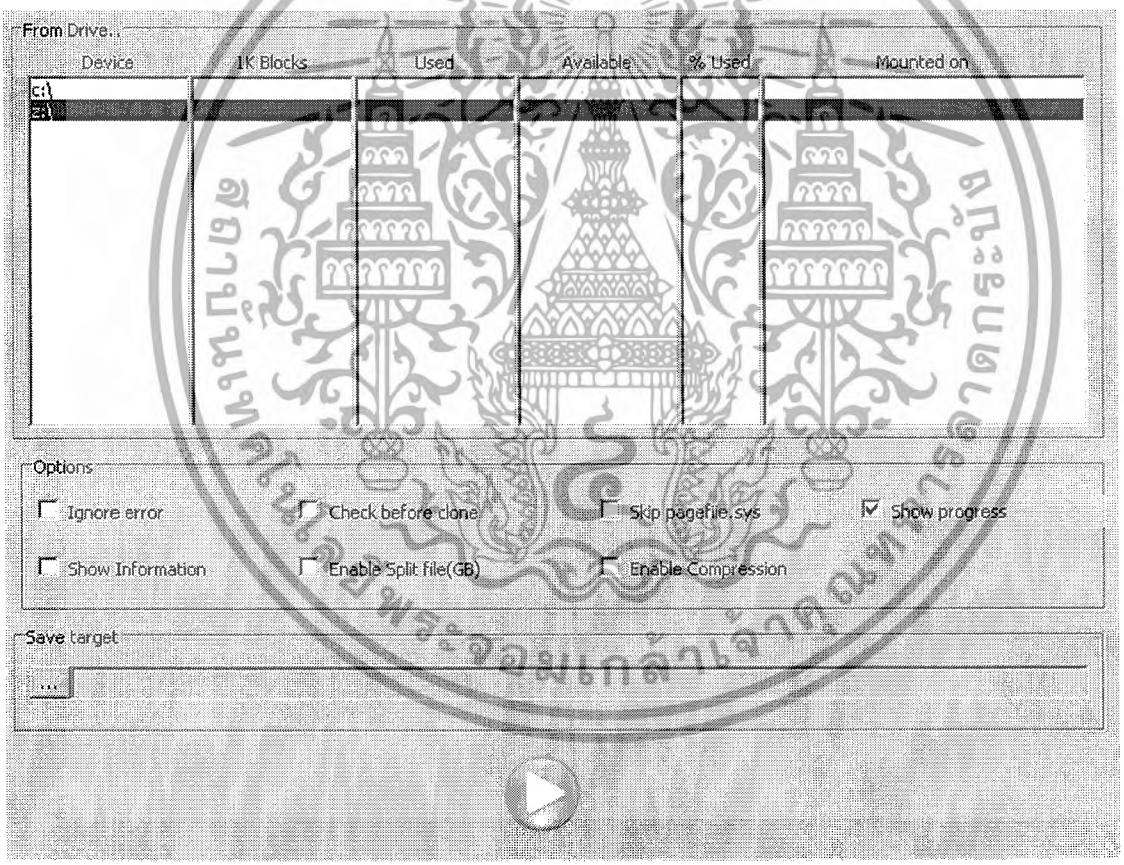
3.1.1 การออกแบบในส่วนของกราฟฟิกโหมด



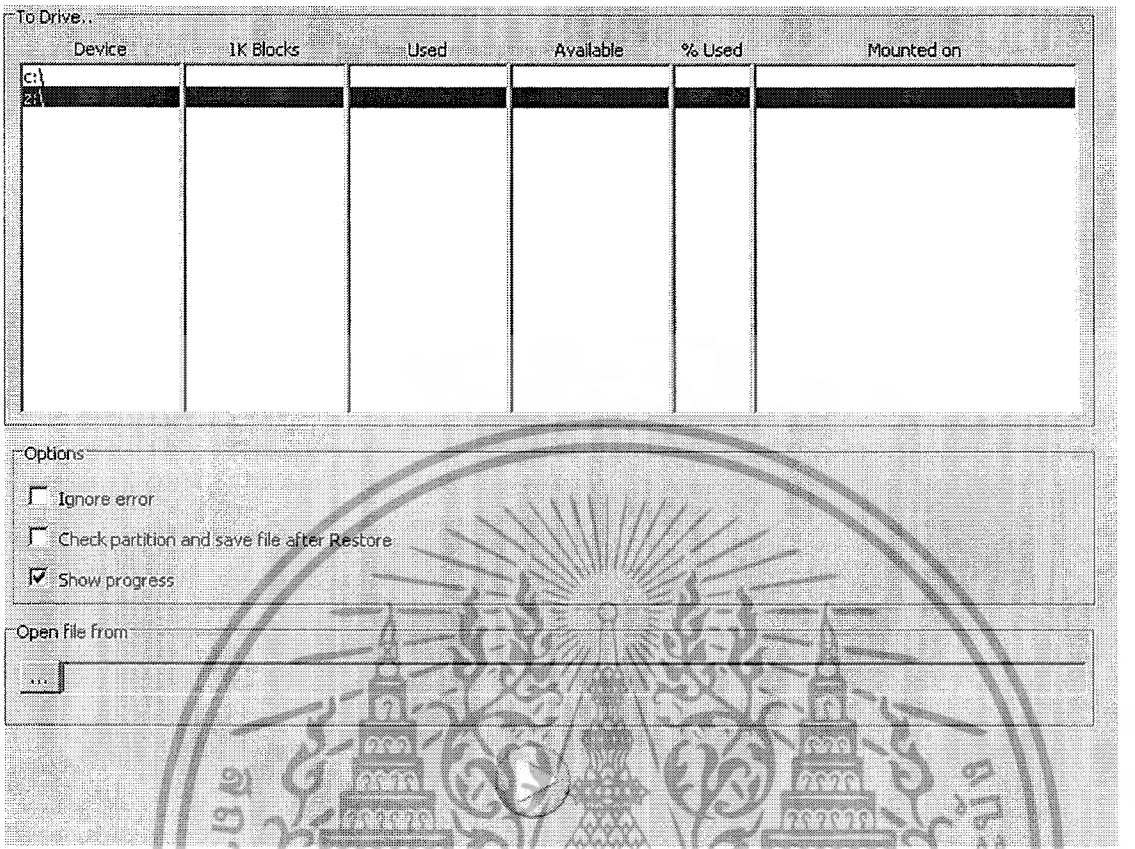
รูปที่ 3.1 แสดงการออกแบบหน้าต่างการทำงานหลัก

ได้ทำการสร้างโดยกำหนดปุ่มเรียกโปรแกรมย่อยภายในกำกับไว้แต่ละปุ่มอีกทอดหนึ่งโดยมี Crack Zip, Undelete , Unformat , Help , Conling , Restore , Compare

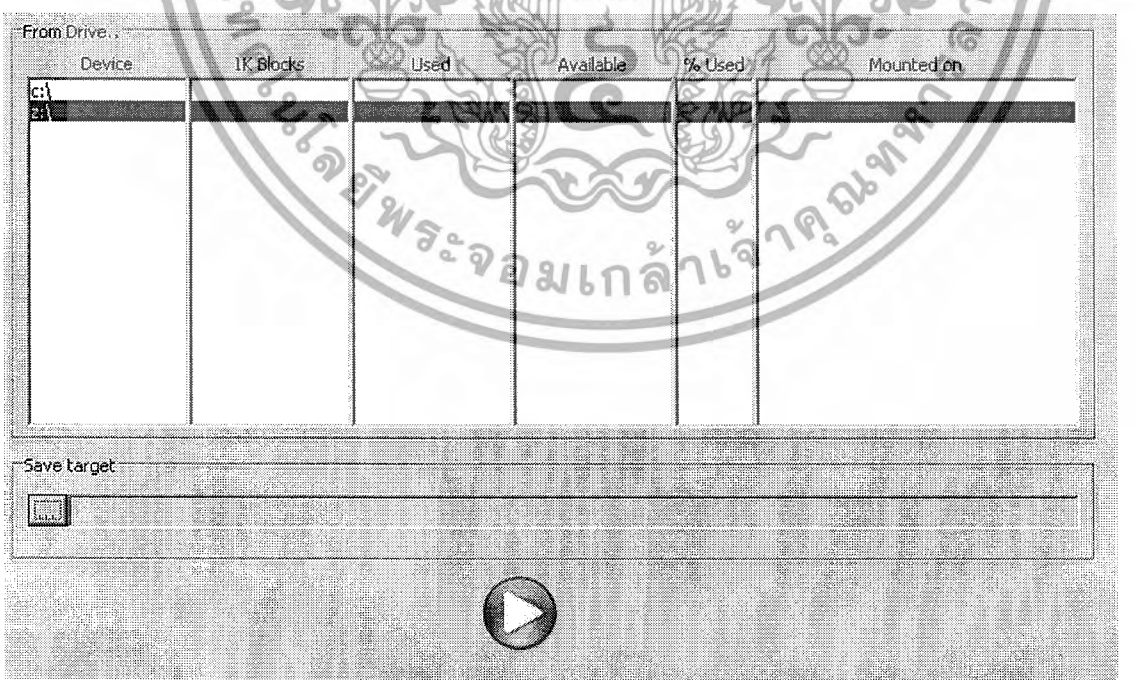
โดยในขั้นตอนนี้ ได้ใช้ภาษา Python ในการพัฒนาส่วนของการแสดงผล โดยเครื่องมือที่ใช้นั้นคือ VisualWx และใช้ Library wxWedget ในการพัฒนาโดยเริ่มจากการติดตั้งโปรแกรม VisualWx ซึ่งสามารถหาดาวน์โหลดมาใช้งานได้ฟรีจากเว็บไซน์ http://visualwx.altervista.org และ Library wxWedget จาก http://www.wxwidgets.org โดยเมื่อทำการติดตั้งโปรแกรมแล้วก็จะสามารถออกแบบส่วนการแสดงผลได้โดยการเลือกวัตถุที่ต้องการจากแถบเครื่องมือ กำหนดฟอร์มรูปแบบตำแหน่ง อีเว้นต์ต่างๆ รวมถึงการนำเข้าคลาส และโค๊ดดิ้งในบางส่วน



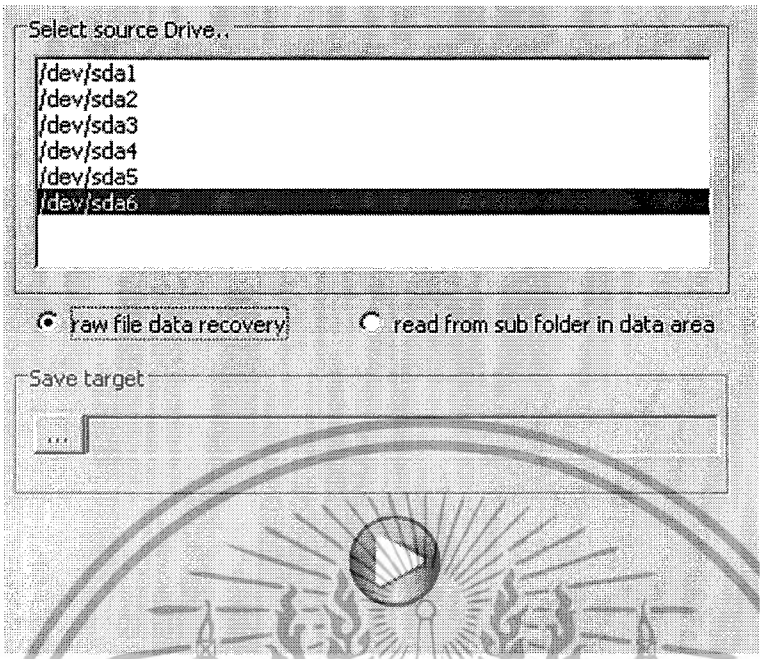
รูปที่ 3.2 แสดงการออกแบบหน้าต่างการ Clone



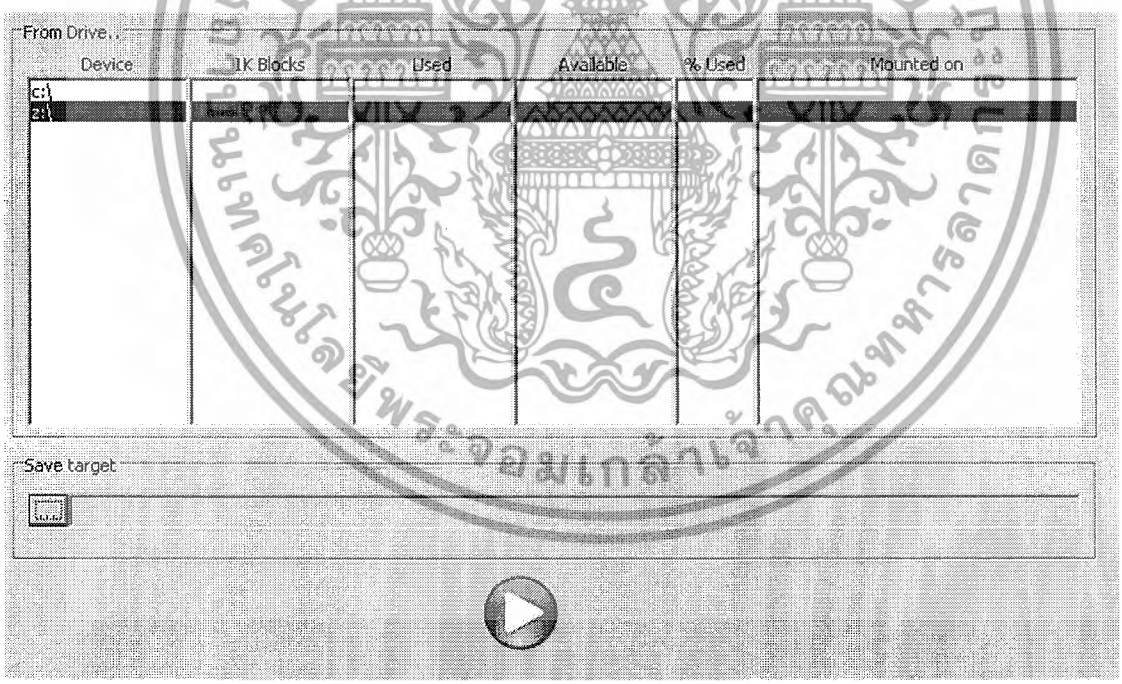
รูปที่ 3.3 แสดงการออกแบบหน้าต่างการ Restore



รูปที่ 3.4 แสดงการออกแบบหน้าต่างการ Undelete



รูปที่ 3.5 แสดงการออกแบบหน้าต่างการ Unformat



รูปที่ 3.6 แสดงการออกแบบหน้าต่างการกู้รหัสผ่าน

เมื่อทำการออกแบบหน้าการแสดงผลแล้วส่วนต่อไปคือการทำการส่งคำสั่งไปไปเรียกใช้การทำงานในแบบคอมมานด์ไลน์อีกครั้งหนึ่ง

3.2 การพัฒนาส่วนของ LiveCD และ การคัดลอกดิสก์

3.2.1 การทำแผ่น LiveCD

Reconstructor คือ เครื่องมือที่พัฒนามาจาก Python ช่วยในการทำแผ่น LiveCD ให้ง่ายขึ้น จุดประสงค์เพื่อให้ผู้ใช้งานนำต้นแบบ (ISO File) ของลินุกส์ที่เบสออน Ubuntu ไม่ว่าจะอยู่ในรูปแบบ Desktop (LiveCD และติดตั้ง) หรือ Alternate (ติดตั้งอย่างเดียว) นำมาดัดแปลงหรือตบแต่งให้เป็นไปลักษณะของผู้จัดทำเองภายในแอพพลิเคชั่นจะมีเครื่องมือช่วยในการตบแต่งหรือดัดแปลง เช่น ภาพพื้นหลัง, ธีมเดสก์ทอป, กรอบหน้าต่าง, และอื่นๆ Reconstructor ได้จัดเตรียมสภาพแวดล้อมรวมถึงเครื่องมือที่ใช้ในการปรับแต่งลินุกส์ตามความต้องการ แต่จะเน้นหนักไปทาง Gnome โดยจะใช้ความสามารถได้เต็มที่ในส่วนของ Desktop Manager ส่วน Linux อื่นๆ เช่น Kubuntu หรือ Xubuntu อาจขาดฟังค์ชั่นไปบ้าง แต่ถ้าหากศึกษาขั้นตอนจัดทำเล็กน้อยก็อาจจะสร้างสคริปต์ขึ้นมาใช้งานได้เอง นอกจากนี้ยังสามารถปรับแต่งในรูปแบบ Command Line ทำให้การปรับแต่งยืดหยุ่นมากขึ้น โดยเครื่องมือสร้าง CD สามารถใช้งานได้กับลินุกส์เบสออน Ubuntu ตั้งแต่รุ่น Dapper > Edgy > Feisty GUI Mode (Frontend) โดยมาก Ubuntu Cristmas / Ubuntu Ultimate / Ubuntu Gamers / Linux Mint ดิสโตรต่างๆเหล่านี้ล้วนถูกปรุงแต่งมาจาก Reconstructor ทั้งสิ้น.

3.2.1.1 ความต้องการของโปรแกรม

- python (only tested on version 2.4)
- pygtk2
- squashfs-tools (needed for Root FS extraction)
- chroot (needed for Root FS customization)
- mkisofs (needed for ISO creation)
- gcc (needed for Usplash generation and VMWare/Qemu module installation)
- make (needed for VMWare/Qemu module installation)
- rsync (needed for Remastering ISO)
- libbogl-dev (needed for Dapper Usplash Generation)
- usplash-dev (needed for Usplash Generation - Edgy and up)
- gnupg (needed for Alternate Key Signing)
- dpkg-dev (needed for Alternate Key Package Building)

- fakeroot (needed for Alternate Key Package Building)
- apt-utils (needed for Extra Repository Generation)

แพ็คเก็ตต่างๆด้านบนหาได้จากแหล่งดาวน์โหลดมาตราฐานโดยทำการเพิ่มเติมส่วนแหล่งดาวน์โหลดเข้าไปใน source.list หากไม่พบ package และหลังจากปรับปรุง source.list แล้วต้องทำการ update source ด้วยคำสั่ง sudo apt-get update

3.2.1.2 การติดตั้งโปรแกรมสร้างแผ่น LiveCD

การติดตั้งโปรแกรมสามารถทำได้โดยการใช้คำสั่งติดตั้งเป็นชุดแพ็คเกจประกอบต่างๆที่เกี่ยวพันกันจะถูกติดตั้งมาด้วย ดังคำสั่งต่อไปนี้ sudo apt-get install squashfs-tools gcc rsync libbogl-dev libusplash-dev gnupg dpkg-dev fakeroot apt-utils เมื่อความต้องการของโปรแกรมถูกติดตั้งเรียบร้อยแล้ว จากนั้นให้ทำการดาวน์โหลดโปรแกรมต้นฉบับจาก http://reconstructor.aperantis.com/index.php?option=com_remository&Itemid=33&func=select&id=5 ในเว็บเพจจะมีชุดโปรแกรมสองแบบให้เลือกติดตั้ง ชุดแรกเป็นซอร์สโค๊ด reconstructor.tar.gz มาคอมไพล์เองตามลักษณะของลินุกส์ ส่วนชุดหลังจะเป็นชุดติดตั้งแบบเดเบี้ยน (deb File) ในที่นี้ใช้แบบ deb file ให้ดาวน์โหลดไฟล์ reconstructor.deb มาที่เครื่องเราแล้วติดตั้งโปรแกรมผ่านเชลล์โปรแกรม ด้วยคำสั่งดังนี้sudo dpkg -i reconstructor.deb เราสามารถเรียกโปรแกรมได้จากระบบเมนู หรือจะเรียกใช้งานผ่านคอมมานด์ไลน์ sudo reconstructor

3.2.2 การทำโปรแกรมคัดลอกดิสก์(Clone & Restore)

การทำโปรแกรมในส่วนนี้จะใช้หลักการคืออ่านแบบไบต์ต่อไบต์ โดยการคัดลอกจะเริ่มจากเซคเตอร์ที่ 0 เป็นต้นไป โดยการทำงานจะประกอบไปด้วยสองส่วนคือ ขั้นตอนการโคลน (clone) และ รีสโตร์(Restore) โดยโปรแกรมต้องมีความสามารถในการรองรับกับการทำงานดังต่อไปนี้

- ตรวจสอบชนิดของ partition
- หาตำแหน่งเริ่มต้นของของอุปกรณ์
- หาขนาดของ partition
- หาขนาดของเซกเตอร์
- หาขนาดของคลัสเตอร์
- หาจำนวนของคลัสเตอร์

- หาขนาดของคลัสเตอร์ทั้งหมด
- ทำการแสกนคลัสเตอร์
- หาตำแหน่งเริ่มของเนื้อข้อมูล
- ทำการสรวจสอบจุดสิ้นสุดของเนื้อข้อมูลเพื่อหาขนาด
- ทำการตรวจสอบจำนวนไฟล์โฟลเดอร์
- ทำการสำเนาชื่อของ partition
- ทำการสำเนาบิตข้อมูลและเขียนลงไฟล์ที่สื่อปลายทาง
- ทำการเปิดอ่านไฟล์ที่บันทึกและทำการเขียนไฟล์ที่อ่านได้กลับไปสู่ partition ตามรูปแบบที่บันทึก
- ทำการทดสอบความถูกต้องนั้นใช้การอ่านไฟล์และ partition ที่เลือกและนำบิตข้อมูลมาเปรียบเทียบกัน

สำหรับการสั่งการให้ทำงานนั้นโปรแกรมที่เขียนต้องสามารถรองรับกับการเลือกใช้พารามิเตอร์ได้โดยจะมีคำสั่ง หลักๆ ที่ใส่เข้าในโปรแกรมเป็นดังนี้

dump	ทำการสำเนาข้อมูล
drydump	ทดสอบการสำเนา
restore	ทำการรีสโตรข้อมูลที่ทำการสำเนาเก็บเป็นไฟล์ไว้กลับคืนสู่ partition
compare	ทำการเปรียบเทียบข้อมูลที่ทำการสำเนาและข้อมูลจริงๆว่าถูกต้องหรือไม่
list	แสดงรายการ partition

โดยรูปแบบการใช้งานที่กำหนดไว้เป็นดังนี้

Dump <command> [option]

โดย option เป็นดังนี้

i	ทำการข้ามส่วนที่อาจจะทำงานผิดพลาด
p	แสดงสถานการณ์ทำงาน
c	ตรวจสอบไฟล์ระบบก่อนที่จะทำการสำเนา (NTFS/FAT32 เท่านั้น)
G	ข้ามการสำเนา pagefiie.sys

d กำหนดอุปกรณ์ที่ต้องการสำเนา

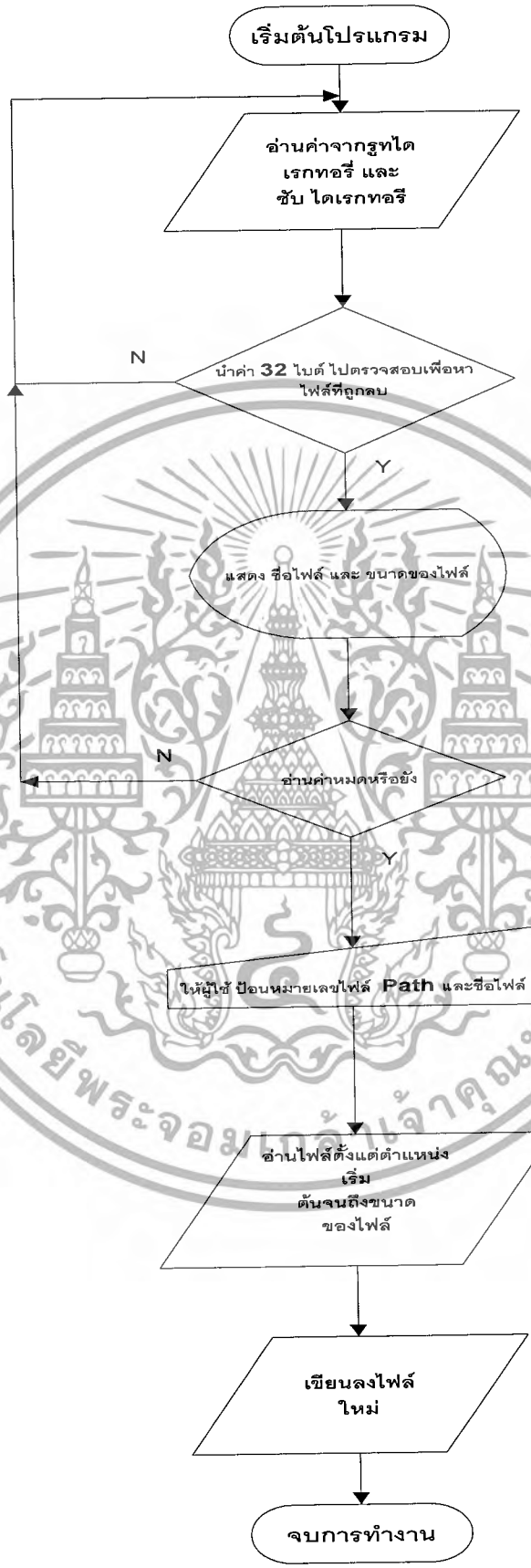
f กำหนดตำแหน่งไฟล์ที่จะบันทึกหรือเรียกคืน

3.3 การพัฒนาส่วนของการกู้ข้อมูล(File Recovery)

3.3.1 โปรแกรม Undelete

ก่อนที่จะทำการเขียนโปรแกรมในการกู้ข้อมูล เราต้องเข้าใจว่าเมื่อลบไฟล์นั้นเกิดอะไรขึ้นกับฮาร์ดดิสก์เราบ้าง เมื่อเราลบไฟล์โดยเมื่อสั่งลบไฟล์ (shift+delete)ระบบปฏิบัติการจะเขียนค่า E5h ลงไปยังไบต์แรกของไดเร็กทอรีที่เก็บชื่อไฟล์ แล้วจัดการล้างข้อมูลใน FAT โดยกำหนดค่า 0 ลงไปตามจำนวนไบต์ของระบบไฟล์นั้น ดังนั้นจะเห็นว่า ถ้าไม่นับตัวอักษรแรก แล้ว การลบแบบนี้จะไม่ได้ทำลายข้อมูลเบื้องต้นของไฟล์เลยแม้แต่น้อย กล่าวคือ ข้อมูลอย่างเช่น หมายเลขคลัสเตอร์แรก ขนาด และ แอตทริบิวต์ ยังอยู่ครบ นอกจากนั้น เนื้อหาของไฟล์ ในพื้นที่จัดเก็บไฟล์ ก็ไม่ได้สูญหายไปไหน ยังคงสภาพเดิมอยู่ทุกประการ เราจึงมีโอกาสที่จะกู้ไฟล์ที่ถูกลบไปคือมาได้ แต่ใช่ว่าทุกไฟล์จะสามารถกู้คืนได้ การจะกู้ไฟล์ให้ได้ครบถ้วนสมบูรณ์ขึ้นอยู่ปัจจัยหลัก 3 ประการ ได้แก่

- หลังจากลบไฟล์แล้ว มีไฟล์หรือไดเร็กทอรี อื่นนำไดเร็คทอรีเอ็นทรีของไฟล์ที่ถูกลบนั้นไปใช้หรือยัง เพราะถ้าปราศจากหมายเลขคลัสเตอร์แรกของไฟล์ ต่อให้เนื้อไฟล์ยังอยู่ครบถ้วนสมบูรณ์ ก็ยากที่จะค้นพบไฟล์นั้นได้
- หลังจากลบไฟล์แล้ว มีไฟล์อื่นมาจับจองใช้งานคลัสเตอร์ของไฟล์ ที่ถูกลบนั้นหรือยัง
- ไฟล์ที่ถูกลบนั้นใช้งานคลัสเตอร์ต่าง ๆ อย่างเป็นลำดับต่อเนื่องหรือไม่



รูปที่ 3.7 แสดงโครงสร้างของโปรแกรม Undelete

ในรูปที่ 3.2 แสดงโครงสร้างของโปรแกรม โดยเริ่มจากการที่เราเข้าไปอ่านค่าจากรูทไดเรกทอรีและซับไดเรกทอรี โดยเราสามารถคำนวณหาตำแหน่งของรูทไดเรกทอรีได้จากค่าที่อ่านได้จากบู๊ตเซกเตอร์ โดยการคำนวณหาค่าตำแหน่งของรูทไดเรกทอรี ได้แก่

*ตำแหน่งของรูทไดเรกทอรี = (ขนาดของบู๊ตเซกเตอร์+ขนาดของแฟต*จำนวนของแฟต+(ขนาดของรูทไดเรกทอรี-2)*จำนวนเซคเตอร์ต่อคลัสเตอร์)*512*

เมื่ออ่านค่าจำนวน 32 ไบต์ในรูทไดเร็กทอรีแล้วเราก็จะนำมาเปรียบเทียบเพื่อหาว่าไฟล์นี้ได้ถูกลบไปแล้วหรือไม่ ถ้าใช่ก็แสดงชื่อไฟล์และขนาดของไฟล์แต่ถ้าไม่ก็จะกลับไปอ่านค่า 32 ไบต์จากรูทไดเรกทอรี อีกครั้งจนหมดทั้งในรูทไดเรกทอรี และซับไดเรกทอรี โปรแกรมก็จะให้ผู้ใช้ป้อนหมายเลขไฟล์ที่ต้องการจะกู้ หลังจากนั้นก็จะให้ผู้ใช้ป้อนไดเรกทอรีปลายทาง พร้อมทั้งชื่อไฟล์ใหม่

เมื่อผู้ใช้ป้อนทุกอย่างครบแล้ว โปรแกรมจะทำการอ่านไฟล์จนเท่ากับขนาดของไฟล์เก่าแล้วนำไปเขียนลงไฟล์ใหม่ที่สร้างขึ้นมา

3.3.2 โปรแกรม Unformat

ในลักษณะของการฟอร์แมตนั้นมีลักษณะที่แตกต่างจากการลบไฟล์อยู่เล็กน้อยคือเมื่อเราสั่งฟอร์แมตไดร์ฟ มันจะทำการเคลียค่าในรูทไดเร็กทอรี่และตาราง FAT เป็น 0 ทั้งหมดซึ่งทำให้เราอ่านจากรูทไดเร็กทอรีไม่ได้เลย มีเพียงในไดเร็กทอรีย่อยเท่านั้นที่ไม่ได้ถูกเคลียด้วย 0 โดยในลักษณะของการกู้ข้อมูลมีอยู่สองลักษณะด้วยกัน

- Row file recovery คือการอ่านข้อมูลทุก ๆ เซ็คเตอร์แล้วหาเฮดเดอร์ของไฟล์ต่าง ๆ ซึ่งในกรณีนี้จะไม่สามารถกู้ไฟล์ได้ทุกนามสกุลของไฟล์เพราะบางไฟล์ก็ไม่มีเฮดเดอร์และฟุตเตอร์ไฟล์
- อ่านค่าจากไดเร็กทอรีย่อย ในวิธีนี้เราจะอ่านทุก ๆ เซคเตอร์เช่นกันเพื่อหาไดเรกทอรี่ย่อยที่ยังหลงเหลือจากการฟอร์แมต

โดยในการเขียนโปรแกรมในส่วนแรก ๆ จะมีความคล้ายคลึงกับโปรแกรม undelete คือจะต้องเข้าไปอ่านค่าต่าง ๆ ในบู๊ตเซคเตอร์ ในโปรแกรมนี้จะให้ผู้ใช้ ใช้งานในสองโหมดด้วยกันคือ

- การหาไฟล์โดยค้นหาไฟล์จากเฮดเดอร์ไฟล์
- การหาไฟล์โดนอ่านค่าจากไดเร็กทอรีย่อย

รูปที่ 3.8 แสดงโครงสร้างของโปรแกรม Unformat ในโหมดค้นหาไฟล์จากเฮดเดอร์

3.3.2.1 ลักษณะโครงสร้างของโปรแกรมในโหมดการค้นหาไฟล์จากเฮดเดอร์ไฟล์

โดยการทำงานในหมวดนี้จะเป็นการอ่านทุก ๆ เซคเตอร์เพื่อหาเฮดเดอร์ของไฟล์และฟุตเตอร์ของไฟล์โดยในโปรแกรมนี้เป็นโปรแกรมตัวอย่างทำงานกับเพียงสองนามสกุลไฟล์เท่านั้นคือ ไฟล์ .doc และไฟล์ .pdf ตัวอย่างเฮดเดอร์และฟุตเตอร์ของทั้งสองไฟล์ได้แก่

PDF_header[] = {0x25,0x50,0x44,0x46,0x2d,0x31,0x2e};

PDF_footer[] = {0x25,0x25,0x45,0x4f,0x46};

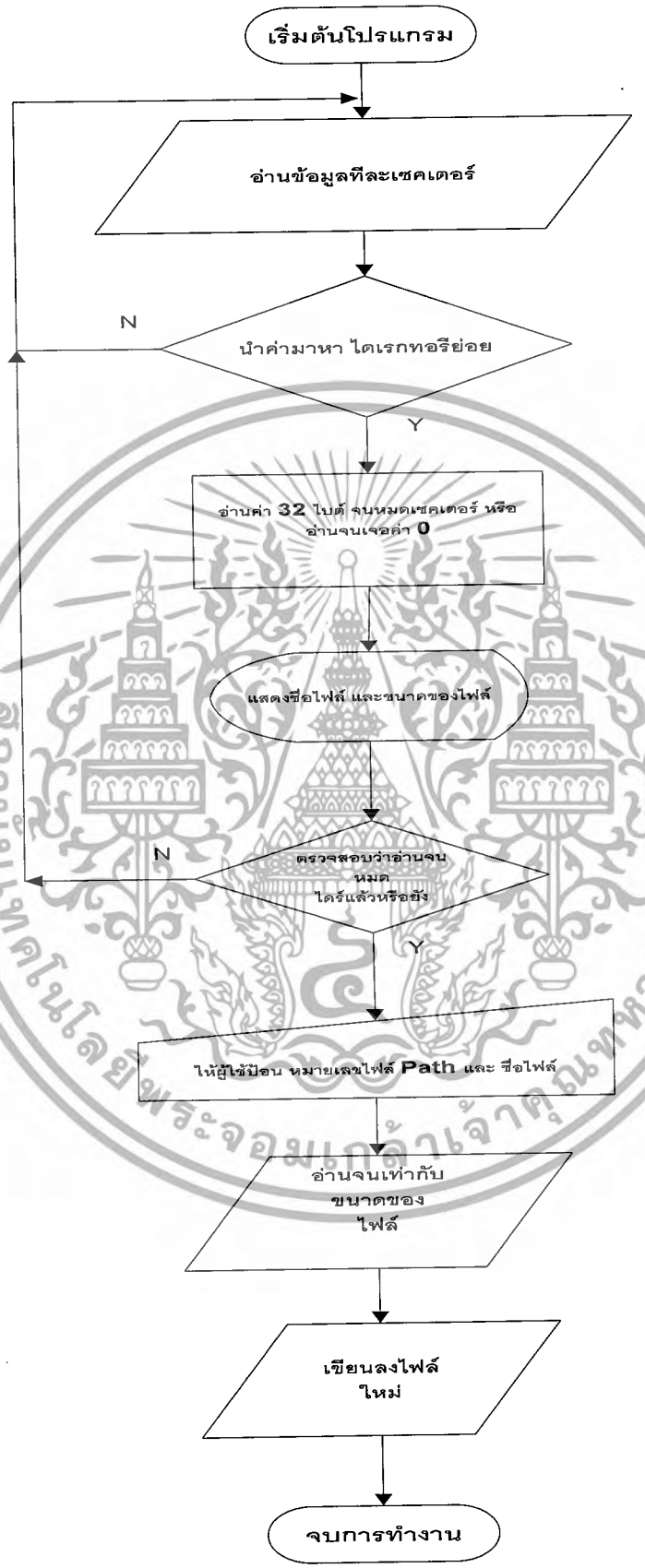
DOC_header[] = {0xd0,0xcf,0x11,0xe0,0xa1,0xb1,0x1a,0xe1};

DOC_footer[]={0x57,0x6f,0x72,0x64,0x2e,0x44,0x6f,0x63,0x75,0x6d,0x65,0x6e,0x74,
0x2e};

เมื่อเรานำเฮดเดอร์ที่มีอยู่ไปเปรียบเทียบกับข้อมูลที่อยู่ในบับเฟอร์ที่เราอ่านได้จากเซคเตอร์ว่าตรงกันแล้วเราก็จะพบนี่คือไฟล์ที่มีนามสกุล .doc อย่างแน่นอน เนื่องจากการหาไฟล์ในลักษณะวิธีนี้เราไม่ได้อ้างอิงชื่อไฟล์จากที่ใด ดังนั้นเราจึงจำเป็นต้องสร้างชื่อไฟล์ขึ้นมาเองเพื่อแสดงให้ผู้ใช้เห็นโดยการเรียงลำดับเลขตั้งแต่ 1 เป็นขึ้นไปซึ่งจะได้เห็นในตัวอย่างผลการทดลองต่อไป

ในส่วนของอ่านและเขียนข้อมูลที่เราต้องการกู้ เราจะใช้หลักการที่ใช้อ่านและเขียนมีอยู่ 2 ประการด้วยกันได้แก่

- อ่านเจอฟุตเตอร์ของไฟล์เดียวกันในกรณีนี้เราจะสมมุติฐานว่าฟุตเตอร์นี้เป็นส่วนหนึ่งของไฟล์นั้นเราก็จะเขียนเป็นเซคเตอร์ที่เจอฟุตเตอร์นี้เป็นเซคเตอร์สุดท้ายของไฟล์
- อ่านเจอฟุตเตอร์ของไฟล์อื่น และเฮดเดอร์ของไฟล์อื่นและของตัวเอง ในกรณีนี้เราจะสมมุติฐานว่าไฟล์นี้ไม่ได้เก็บแบบเรียงลำดับเราจะจบการอ่านไฟล์นี้ทันที โดยไม่เขียนเซคเตอร์เหล่านี้ลงไปด้วย



รูปที่ 3.9 โครงสร้างของโปรแกรม Unformat ในโหมดไดเรกทอรีย่อย

3.3.2.2โครงสร้างของโปรแกรมในโหมดการค้นหาไฟล์จากการอ่านไดเรกทอรีย่อย

โหมดที่สอง เป็นโหมดที่ใช้ไดเร็กทอรีย่อยให้เป็นประโยชน์เพราะเมื่อเราฟอร์แมตไดร์ฟข้อมูลในไดเร็กทอรีย่อยจะไม่ถูกลบไปด้วย ซึ่งกระบวนการการกู้ข้อมูลก็จะคล้าย ๆ กับกระบวนการของการ Undelete เพราะเมื่อเราทราบข้อมูลที่มีอยู่ในไดเร็กทอรีย่อยแล้ว เราก็สามารถที่จะทราบว่าไฟล์นั้น ๆ ใช้จำนวนคลัสเตอร์ในการเก็บไฟล์เป็นจำนวนกี่คลัสเตอร์ แต่กระบวนการนี้อาจทำให้เราอ่านข้อมูลผิดไฟล์ได้เช่นกันถ้าหากว่าไฟล์ที่เราต้องการจะกู้นั้นถูกเขียนทับด้วยไฟล์อื่นไปแล้ว ซึ่งในส่วนนี้ยังเป็นข้อจำกัดของโปรแกรมนี้อยู่

3.4 การพัฒนาส่วนของการกู้รหัสผ่านไฟล์ Zip/Rar/7z

โปรแกรมจะทำการเปิดไฟล์ที่บีบอัดขึ้นมาแล้วทำการตรวจเฮดเดอร์เพื่ออ่านชนิดของไฟล์ จากนั้นจึงทำการเอาชนิดไฟล์ที่ได้ไปทำการเลือกโปรแกรมที่จะทำการแกะอีกครั้ง โดยค่าเฮดเดอร์จะมีลักษณะดังนี้

rar = application/x-rar

zip = application/zip

7z = application/octet-stream

โดยการแกะรหัสนั้นจะใช้วิธีการส่งพารามิเตอร์เข้าไปทำงานด้วยโปรแกรม unrar , unzip , 7z ซึ่งตัวโปรแกรมที่มีให้ใช้อยู่แล้วใน LiveCD โดยในการสั่งงานของ unrar นั้นใช้รูปแบบดังนี้ unrar t –p รหัสผ่าน ชื่อและนามสกุลไฟล์โดย t หมายถึงการสั่งทำการทดสอบในส่วนของ unzip จะใส่พารามิเตอร์ unzip t –p รหัสผ่าน ชื่อไฟล์ zip และในส่วนของ 7z จะเป็น 7z t –p รหัสผ่าน ชื่อไฟล์ 7z ซึ่งเมื่อทำการเขียนเป็นคำสั่งแล้วจะนำไปรันใน popen() เพื่อให้ทำงานโดยทำการแตกเทรดตามจำนวนที่กำหนดและในระหว่างที่โปรแกรมทำงานนั้นก็จะทำการรับส่งข้อมูลที่ ทำการทำงานไปแล้วผ่าน Pipe อีกทีหนึ่งเพื่อตรวจดูว่าผลลัพธ์การทำงานนั้นสำเร็จหรือไม่ โดยการสแกนส่วนของผลลัพธ์ที่มีข้อความ ok ปรากฏอยู่ซึ่งโปรแกรมก็จะทำการทำงานไปเรื่อยๆจนกระทั้งพบรหัสผ่านและแสดงผลลัพธ์ออกมาแสดงผล โดยกระบวนการในการสร้างรหัสผ่านนั้นก็จะมีสองส่วนคือ brute force และ dictionary โดย brute force นั้นจะทำการกำหนดช่วงของรหัสตัวอักษรที่จะเป็นไปได้ไว้ เช่น 0-9 , a-z ,A-Z หรืออักขระพิเศษอื่นๆที่ต้องการและจากนั้นก็ทำการอ่านรหัสตัวอักษรออกมาเพื่อสร้างรหัสผ่านตามจำนวนหลักที่ต้องการและนำไปทำการถอดรหัสต่อไป

และในส่วนของ dictionary นั้นก็เพียงทำการเปิดไฟล์ที่เป็น text dictionary และทำการอ่านไปเรื่อยๆจนหมดไฟล์ โดยแต่ละคำที่ได้นั้นก็จะนำไปแกะรหัสผ่านโดยตรงเช่นกัน

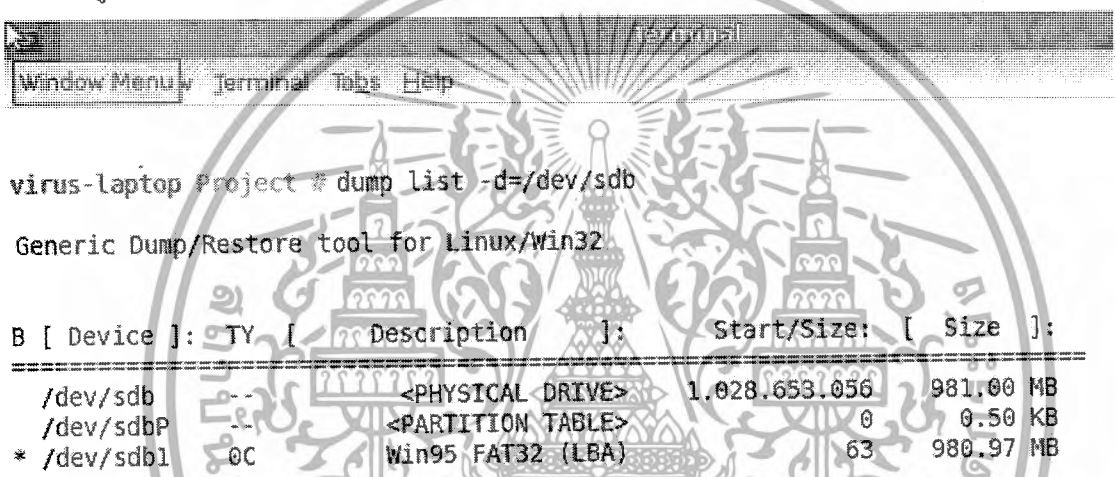
ซึ่งกระบวนการทำงานทั้งหมดนั้นโปรแกรมได้ออกแบบให้สามารถทำการเก็บสถานะไว้เป็นไฟล์ว่าขณะนั้นทำงานไปได้ถึงอักขระที่เท่าไหร่แล้วดังนั้นจึงสามารถหยุดการทำงานได้และกลับมาทำงานใหม่อีกครั้ง โปรแกรมก็จะทำการอ่านไฟล์ หรือสร้างรหัสผ่านจากส่วนล่าสุดที่เคยบันทึกไว้

บทที่ 4

การทดลองและผลการทดลอง

4.1 การทดลองคัดลอกดิสก์

การพัฒนาเรื่องการสำเนาข้อมูลโดยได้ทำการทดลองสำเนาแฟลชไดร์ฟ ออกมาเป็นไฟล์ข้อมูลโดยมีรายละเอียดการแสดงดังนี้



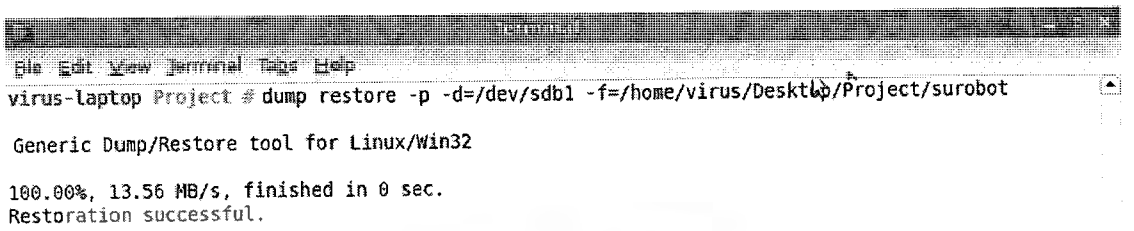
รูปที่ 4.1 แสดงหน้าจอผลการแสดง partition แบบคอมมานไลน์

โดยไดร์ฟที่ทำการทดลองคัดลอกจะเป็น /dev/sdb1 ซึ่งเป็นชนิด FAT32 ขนาด ~1GB โดยก่อนทำการทดสอบภายในมีข้อมูลคร่าวๆดังนี้

```
virus-laptop Project # ls -l /media/SUROBOT
total 5712
-rwx------ 1 virus root    5876 2007-05-25 19:41 autoexec.bat
-rwx------ 1 virus root     844 2006-10-04 10:37 autoexec.dat
drwx------ 4 virus root    4096 2008-11-08 09:54 BootCD
-rwx------ 1 virus root     246 2004-05-30 22:21 BootDriv.exe
-rwx------ 1 virus root      78 2008-12-01 17:07 BOOTLOG.PRV
-rwx------ 1 virus root      78 2008-12-07 17:10 BOOTLOG.TXT
-rwx------ 1 virus root     516 2008-11-11 17:09 BOOT.SAV
-r-x------ 1 virus root   94292 2003-05-05 22:22 COMMAND.COM
-rwx------ 1 virus root   15549 2007-05-27 22:37 config.sys
-rwx------ 1 virus root    2989 2005-11-06 23:08 devload.com
-rwx------ 1 virus root  243627 2006-09-05 15:42 docmem.exe
```

รูปที่ 4.2แสดงหน้าจอผลการแสดง รายละเอียดไฟล์ในสื่อ

โดยในครั้งนี้ได้ทำการทดลองกู้คืนข้อมูลที่ได้ทำการคัดลอก ไว้กลับคืนโดยใช้คำสั่งดังนี้



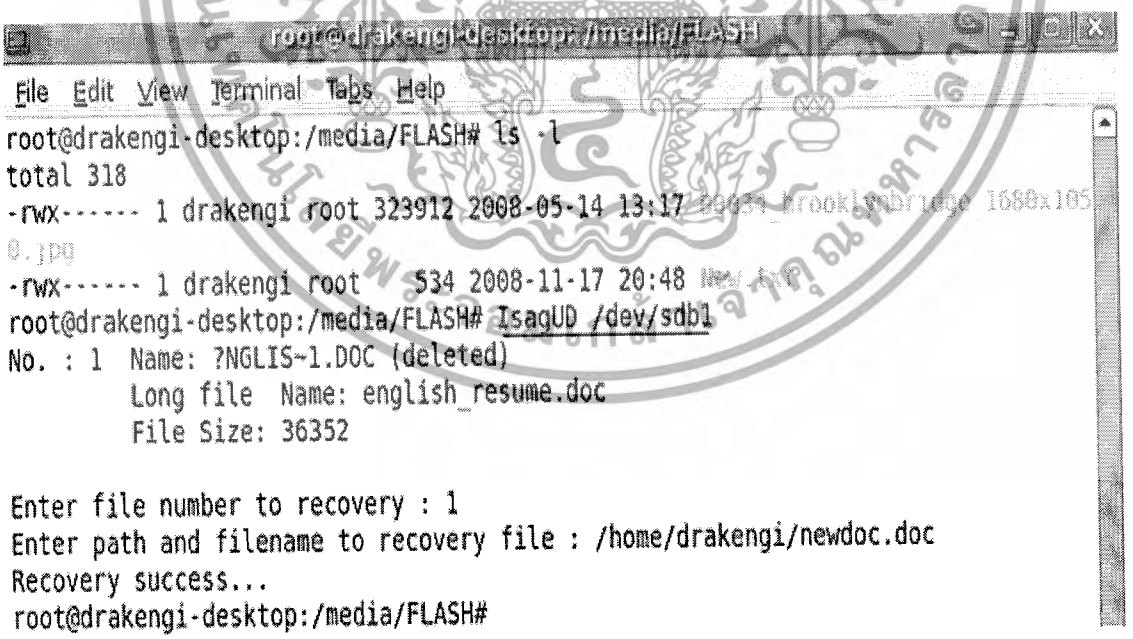
รูปที่ 4.3 แสดงหน้าจอผลการทำงานของการ Clone แบบคอมมานไลน์

โดยในการทดสอบหลังจากที่ทำการคัดลอกแล้ว ได้ทำการฟอร์แมต แฟลชไดร์ฟ ก่อนแล้วจึงทำการกู้คืนกลับซึ่งก็พบว่าข้อมูลที่ได้สามารถกลับคืนมาได้ปกติดี

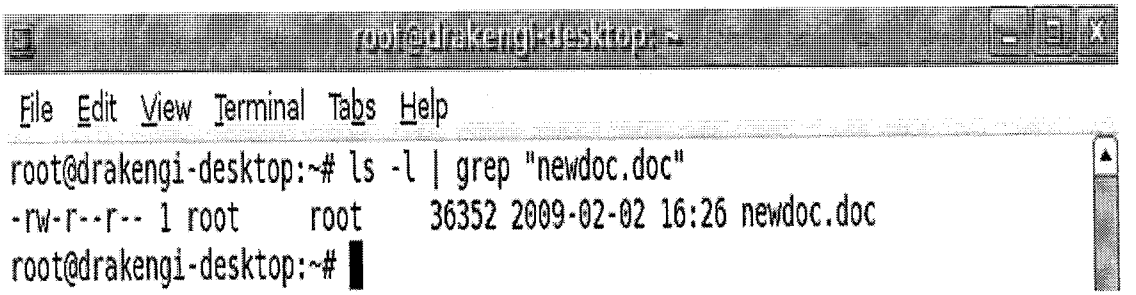
4.2 การทดลองการกู้ไฟล์

4.2.1 การทดลองโปรแกรม Undelete

การทดลองโปรแกรม undelete เราจะใช้ ทรัมไดร์(trumb drive) ในการทดลองเพราะเนื่องจากมีขนาดเล็ก และเป็นระบบไฟล์แบบ FAT32 จึงเหมาะที่จะใช้ในการทดลอง



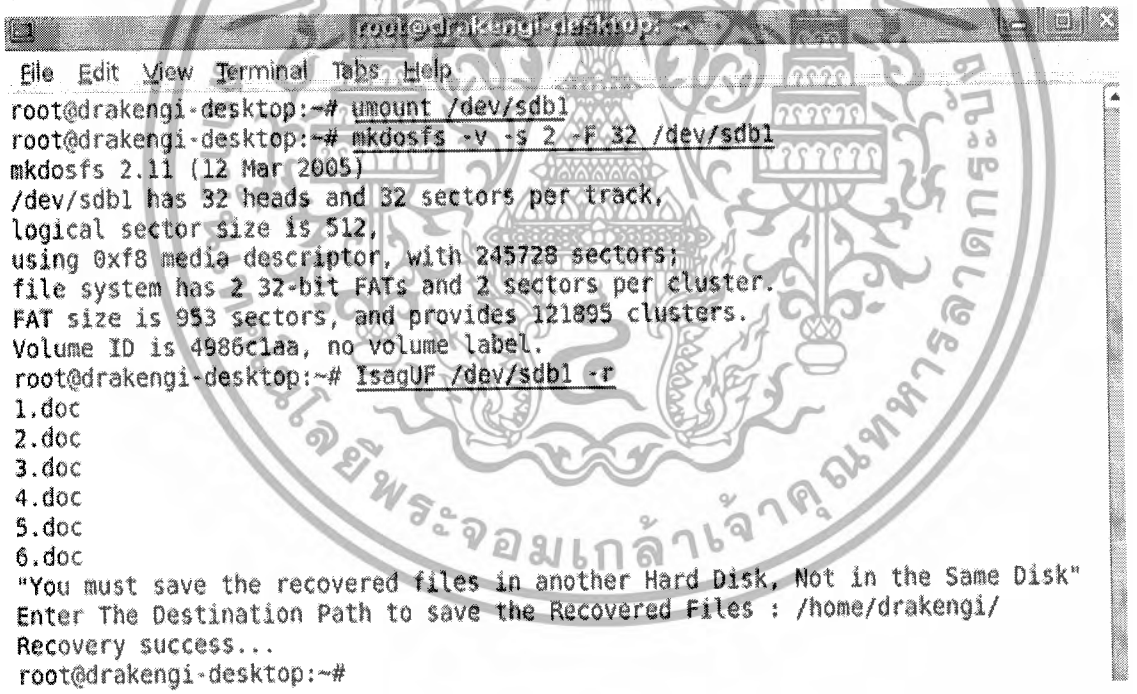
รูปที่ 4.4 แสดงการทำงานของโปรแกรม Undelete แบบคอมมานไลน์



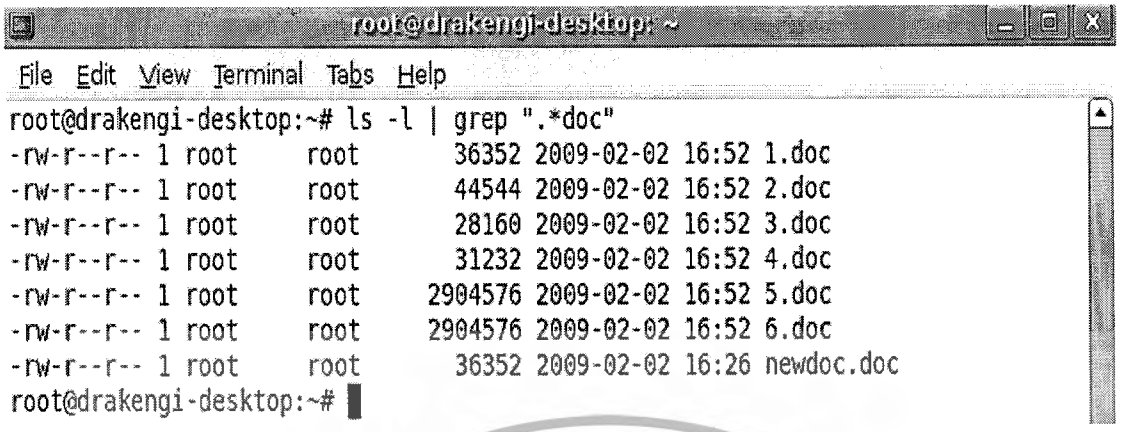
รูปที่ 4.5 แสดงผลลัพธ์ของโปรแกรม Undelete

4.2.2 การทดลองโปรแกรม Unformat

ในการทดลองโปรแกรม Unformat นี้เราจะใช้ทรัมไดร์ในการทดลองเช่นกันโดยการทดลอง เราก็จะต้องทำการฟอร์แมตทรัมไดร์เสียก่อน ด้วยคำสั่ง mkdosfs ดังเช่นในรูปที่ 4.6 ซึ่งในรูปที่ 4.6 เป็นการทดลองในโหมดการหาไฟล์จากเฮดเดอร์ไฟล์

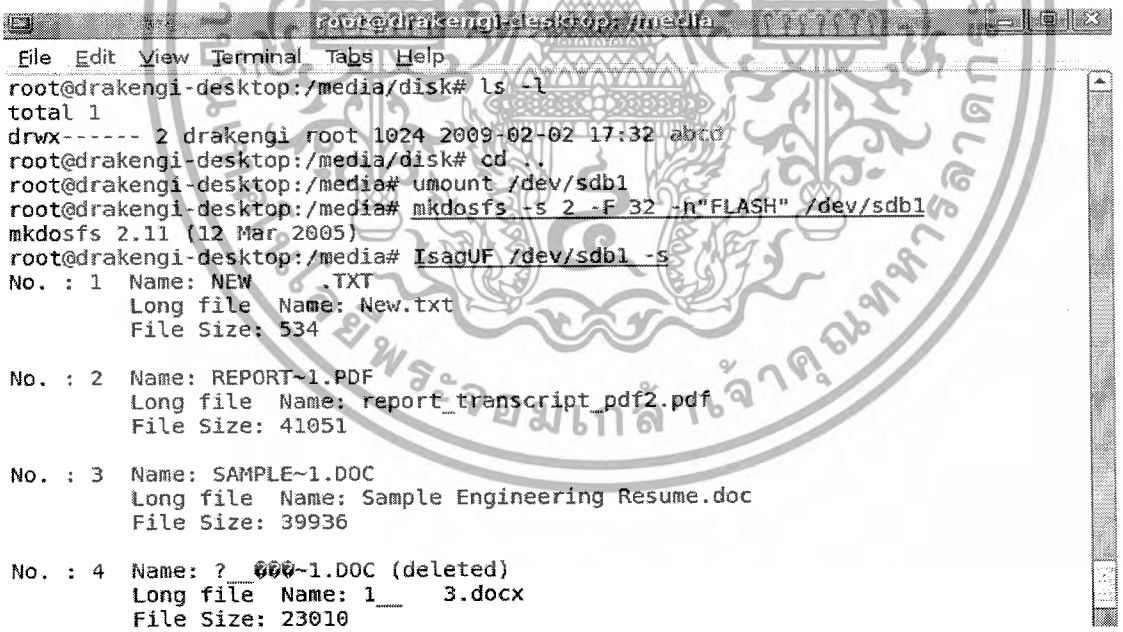


รูปที่ 4.6 แสดงการทำงานของโปรแกรม Unformat ในโหมด <-r>



รูปที่ 4.7 แสดงผลลัพธ์ของโปรแกรม Unformat ในโหมด <-r>

ส่วนการใช้งานในโหมด < -s > นั้นผลที่ได้อาจแตกต่างกับโหมดแรกอยู่พอสมควรเพราะการทำงานไม่ได้อ่านเซคเตอร์เพื่อหาเฮดเดอร์ไฟล์แต่จะเป็นการอ่านแต่ละเซคเตอร์เพื่อหาไดเรกทอรีย่อยซึ่งผลการรันจะได้ออกมาดังเช่นในรูปที่ 4.8



รูปที่ 4.8 แสดงการทำงานของโปรแกรม Unformat ในโหมด <-s>

```
please enter number of file to be recovery : 1
"You must save the recovered files in another Hard Disk, Not in the Same Disk"
Enter The Destination Path and filename to save the Recovered Files : /home/drak
engi/aa.txt
Recovery success...
root@drakengi-desktop:/media#
```

รูปที่ 4.9 แสดงผลลัพธ์ของโปรแกรม Undelete ในโหมด <-s>

4.3 การทดลองกู้รหัสพาสเวิร์ดไฟล์ Zip/Rar/7z

การพัฒนาโปรแกรมแกะรหัสผ่านในส่วนนี้นั้นใช้กรรมวิธีการ brute force หรือ dictionary เข้าไปแกะรหัสผ่านและนอกจากนั้นยังทำการเทรดออกเป็นส่วนย่อยๆเพื่อเพิ่มความเร็วในการทำงานขึ้นโดยคำสั่งพื้นฐานที่ใช้จะเป็นดังนี้

รูปที่ 4.10 แสดงโปรแกรม ForensicCrack แบบคอมมานไลน์

จากนั้นทำการทดลองแกะรหัสผ่านขนาด 2 หลักโดยใช้ตัวเลขอย่างเดียวโดยใช้รหัส 99

```
virus-desktop zip # forensicCrack ./n2.zip
Compression Crack

INFO: detected file type: zip
INFO: cracking ./n2.zip, status file: ./n2.zip.xml
GOOD: password cracked: '99'
virus-desktop zip #
```

รูปที่ 4.11 แสดงผลลัพธ์ของการแกะรหัสขนาด 2 หลักตัวเลขเดี่ยว

ผลลัพธ์ของการแกะรหัสผ่านใช้เวลาแค่ประมาณ 3 วินาที จากนั้นจึงทำการทดสอบรหัสผ่านขนาด 2 หลักแต่มีตัวอักษรผสมอยู่ด้วยโดยใช้รหัส Z9

```
virus-desktop zip # forensicCrack ./ns2.zip
Compression Crack

INFO: detected file type: zip
INFO: cracking ./ns2.zip, status file: ./ns2.zip.xml
Probing: 'b9' [250 pwds/sec]
Probing: 'nh' [250 pwds/sec]
Probing: 'zl' [249 pwds/sec]
Probing: 'Lq' [249 pwds/sec]
Probing: 'Xv' [249 pwds/sec]
GOOD: password cracked: 'Z9'
virus-desktop zip #
```

รูปที่ 4.12 แสดงผลลัพธ์ของการแกะรหัสขนาด 2 หลักตัวอักษรผสมตัวเลข

ผลการทดสอบพบว่าระยะเวลาที่ใช้นั้นนานขึ้นเป็น 17 วินาทีเนื่องจากความเป็นไปได้ที่จะต้องทำการสุ่มรหัสผ่านนั้นมากขึ้น จากนั้นทำการทดสอบแกะรหัสผ่านขนาด 3 หลักโดยใช้ตัวเลขอย่างเดียวใช้รหัส 999

```
virus-desktop zip # forensicCrack ./n3.zip
Compression Crack

INFO: detected file type: zip
INFO: cracking ./n3.zip, status file: ./n3.zip.xml
Probing: 'aL' [242 pwds/sec]
Probing: 'mv' [242 pwds/sec]
Probing: 'yf' [242 pwds/sec]
Probing: 'JW' [241 pwds/sec]
Probing: 'VE' [242 pwds/sec]
Probing: '076' [236 pwds/sec]
Probing: '0iP' [242 pwds/sec]
Probing: '7i6' [241 pwds/sec]
Probing: '7tV' [244 pwds/sec]
Probing: '7FA' [241 pwds/sec]
Probing: '7Rh' [241 pwds/sec]
Probing: '82X' [241 pwds/sec]
Probing: '8eE' [241 pwds/sec]
Probing: '8qp' [243 pwds/sec]
Probing: '8Cf' [244 pwds/sec]
Probing: '8Ma' [205 pwds/sec]
Probing: '8U4' [163 pwds/sec]
Probing: '923' [165 pwds/sec]
GOOD: password cracked: '999'
virus-desktop zip #
```

รูปที่ 4.13 แสดงผลลัพธ์ของการแกะรหัสขนาด 3 หลักตัวเลขเดี่ยว

ผลการทดสอบพบว่าระยะเวลาที่ใช้นั้นนานขึ้นเป็น 2.47 นาที เนื่องจากความเป็นไปได้ที่จะต้องทำการสุ่มมากขึ้นนั่นเอง จากนั้นได้ทำการทดสอบขนาด 3 หลักโดยใช้ตัวเลขอย่างเดียวใช้รหัส 999 โดยการจัดการกับเทรดเข้าช่วยโดยเพิ่มจำนวนเป็น 5 ผลลัพธ์ที่ได้พบว่าใช้ความเร็วเหลือประมาณ 2.25 นาที

```
virus-desktop zip # forensicCrack ./n3.zip --threads 5
Compression Crack

INFO: detected file type: zip
INFO: cracking ./n3.zip, status file: ./n3.zip.xml
Probing: 'b7' [248 pwds/sec]
Probing: 'mU' [243 pwds/sec]
Probing: 'xj' [215 pwds/sec]
Probing: 'J6' [243 pwds/sec]
Probing: 'UN' [241 pwds/sec]
Probing: 'O6A' [243 pwds/sec]
Probing: 'Oil' [243 pwds/sec]
Probing: 'Ou6' [243 pwds/sec]
Probing: 'OFS' [243 pwds/sec]
Probing: 'ORB' [242 pwds/sec]
Probing: '13q' [244 pwds/sec]
Probing: '1fb' [243 pwds/sec]
Probing: '1qY' [243 pwds/sec]
Probing: '1CN' [244 pwds/sec]
Probing: '1Ou' [241 pwds/sec]
Probing: '207' [240 pwds/sec]
Probing: '2bW' [244 pwds/sec]
Probing: '2mN' [245 pwds/sec]
```

รูปที่ 4.14 แสดงผลลัพธ์ของการแกะรหัสขนาด 3 หลักตัวเลขเดี่ยวและใช้เทรดเข้าช่วย

จากนั้นได้ทำการทดสอบขนาด 3 หลักโดยใช้ตัวเลขอย่างเดียวใช้รหัส 999 โดยการจัดการกับเทรดเข้าช่วยโดยเพิ่มจำนวนเป็น 20 ผลลัพธ์ที่ได้พบว่ากลับพบว่าช้ากว่าเดิม โดยใช้ความเร็วประมาณ 3.15 นาทีซึ่งจากการทดสอบและสังเกตุการณ์พบว่าความเร็วที่ลดลงนั้นส่วนหนึ่งเกิดมาจากการที่ระบบต้องทำการเสียเวลาไปกับการเขียนไฟล์ผลลัพธ์ลง xml ซึ่งจำเป็นต้องติดต่อกับ HDD ซึ่งทำให้ความเร็วที่ได้จะไปหยุดอยู่แค่จุดที่จะทำการอ่านเขียนได้เร็วที่สุดเท่านั้นจึงทำให้การเพิ่มเทรดขึ้นมาช่วยนั้นไม่ค่อยมีผลมากนัก

บทที่ 5

บทสรุปและ วิจารณ์

5.1 บทสรุปและวิจารณ์

โดยสรุปการทำงานของโครงงานนี้เราได้มาซึ่ง LiveCD และโปรแกรมต่าง ๆ ที่ใช้สำหรับเก็บหลักฐาน ซึ่งสามารถรันอยู่บน LiveCD ได้เช่น

5.1.1 โปรแกรม Clone&Restore โปรแกรมนี้ทำให้เราสามารถที่จะสำเนาข้อมูลในฮาร์ดดิสก์ แฟลชเมโมรี ฯ ของเครื่องเป้าหมาย โดยไม่จำเป็นต้องเข้าไปยุ่งเกี่ยวกับระบบปฏิบัติการของเครื่อง และเราไม่จำเป็นที่จะต้องนำฮาร์ดดิสก์ ของเครื่องเป้าหมายมาเพื่อเก็บไว้เป็นหลักฐาน

5.1.2 โปรแกรม Undelete/Unformat ในกรณีที่เครื่องเป้าหมายนั้นได้ถูกลบหรือทำลายหลักฐาน เราจะสามารถใช้โปรแกรม Undelete/Unformat เพื่อที่จะกู้ข้อมูลที่อาจใช้เป็นหลักฐานกลับมาได้

5.1.3 โปรแกรม Crack Zip,Rar,7z ในกรณีที่เราพบไฟล์ที่ถูกบีบอัดพร้อมทั้งยังเข้ารหัสไว้ด้วย เราสามารถที่จะใช้โปรแกรม crack ในการค้นหารหัส และนำไฟล์นั้นไปประมวลผลยังส่วนต่อไป

5.2 ปัญหาและอุปสรรค

ในระหว่างกระบวนการพัฒนา LiveCD นั้นจะประสบปัญหาบ้างอยู่นั่นคือเรื่องของการรวบรวมไฟล์สำหรับการติดตั้งเพื่อสร้างแผ่น LiveCD เนื่องจากแหล่งข้อมูลในการพัฒนานั้นจะมาจากอินเตอร์เน็ทเป็นหลักเนื่องจากว่ายังไม่มีเอกสารในการพัฒนาเป็นตัวอย่างที่ชัดเจน ดังนั้นเมื่อทำการศึกษาเพื่อพัฒนาจะพบถึงความหลากหลาย เนื่องจากว่าตัว Linux เองจะมีเวอร์ชั่นใหม่ๆที่ออกมาเรื่อยๆ ดังนั้นในขณะที่ทำการพัฒนานั้นไฟล์หรือแหล่งข้อมูลอ้างอิงจะอ้างอิงถึงเวอร์ชั่นที่ผู้ทำการพัฒนากำลังทำงานอยู่ด้วยเท่านั้น

ส่วนของแอพพลิเคชั่นที่ใช้สำหรับการสืบค้น กู้คืน และแกะรหัสผ่านข้อมูล ในขั้นตอนของการพัฒนานั้น นอกจากจะต้องทำแอพพลิเคชั่นให้ทำงานได้ตามวัตถุประสงค์ที่ต้องการแล้ว เรา

จะต้องทำให้แอพพลิเคชั่นนี้ทำงานร่วมกับ LiveCD ได้และนอกจากนั้นยังต้องสามารถที่จะทำงานได้ตามจุดประสงค์ของโครงงานที่ตั้งไว้อีกด้วย

5.3 แนวทางการพัฒนาต่อ

5.3.1 การพัฒนาส่วนของแผ่น LiveCD เนื่องจากการทำงานของ LiveCD นั้นได้ทำการติดตั้งเครื่องมือรวมทั้งโหลดโปรแกรมต่างๆเข้าไปด้วยดังนั้นหากผู้พัฒนาต้องการปรับแต่งแก้ไขหรือเพิ่มเติมผู้พัฒนาสามารถทำการปรับแต่งได้ตามต้องการเนื่องจากเครื่องมือที่ใช้ในการออกแบบระบบนั้นอนุญาตให้ทำการแก้ไขอย่างไรก็ได้

5.3.2 การพัฒนาส่วนของการ Clone & Restore ต้องทำการปรับปรุงแก้ไขในเรื่องของความสามารถในการทำงานส่วนอื่นๆเช่นรองรับการบีบอัดไฟล์ให้มีขนาดเล็กลงได้ดียิ่งขึ้น

5.3.3 พัฒนาการปรับปรุงโค๊ดโปรแกรมทั้ง Undelete และ Unformat เพื่อให้ทำงานกับดิสก์ขนาดใหญ่ได้รวดเร็วขึ้น ด้วยการเขียนแบบหลายโปรเซส หรือ แตกเทรด เพื่อให้สามารถทำการประมวลผลได้เร็วขึ้น รวมทั้งเรื่องของการดักจับข้อผิดพลาด(error) และพัฒนาให้สามารถตรวจสอบไฟล์ที่จะกู้ข้อมูลก่อนที่จะทำการกู้ข้อมูลได้ด้วย

5.3.4 การพัฒนาส่วนของการแกะรหัสผ่านนั้นควรจะเพิ่มรูปแบบของไฟล์ที่รองรับให้มากยิ่งขึ้น และทำงานเรื่องของการเพิ่มความเร็วโดยการแก้ไขปัญหาการติดต่อกับไฟล์เก็บสถานะ และใช้การเขียนสถานะลงเมมโมรี่แทนเพื่อช่วยการทำงาน

บรรณานุกรม

พร้อมเลิศ หล่อวิจิตร. 2545. **ผ่า! Hard disk.** กรุงเทพฯ : โปรวิชั่น.

Reconstructor is an Ubuntu GNU/Linux CD Creator ,

[online][URL] http://www.easyzonecorp.net/network/view.php?ID=660

Reconstructor,

[online][URL] http://reconstructor.aperantis.com

Remastersys,

[online][URL] http://www.remastersys.klikit-linux.com

LiveCD,

[online][URL] http://livecd.org

Recovery data,

[online][URL] http://www.p-dd.com/data-recovery-programming-book.html

FAT32,

[online][URL] http://en.wikipedia.org/wiki/File_Allocation_Table

Python,

[online][URL] http://www.python.org

wxPython,

[online][URL] http://www.wxpython.org

VisualWx,

[online][URL] http://visualwx.altervista.org

WingIDE,

[online][URL] http://www.wingware.com

Sample Code,

[online][URL] http://sourceforce.net,[URL] http://google.com/codesearch