บทที่ 2
หลักการและทฤษฎี

จากบทที่หนึ่งได้กล่าวถึงความสำคัญของการทำโครงงานและรูปแบบขั้นตอนการดำเนินงาน แต่ในการที่จะดำเนินโครงการนี้ต่อไปได้ จำเป็นที่จะต้องอาศัยความรู้ความเข้าใจพื้นฐานเกี่ยวกับ การทำงานของเว็ปเซิร์ฟเวอร์ (Web Server) ระบบปฏิบัติการลีนุกซ์ (Linux) และ ระบบปฏิบัติการยูนิกซ์ (UNIX) เพื่อใช้ในการพัฒนาและบริหารจัดการเว็ปเซิร์ฟเวอร์ของระบบเครือข่ายคอมพิวเตอร์ภายในคณะวิศวกรรมศาสตร์

2.1 Web Server

ในยุคอินเตอร์เน็ตอย่างทุกวันนี้ เว็ปเซิร์ฟเวอร์เป็นกลไกสำคัญต่อองค์กรหลายแห่ง และช่วยสนับสนุนแอปพลิเคชั่น (Application) ใหม่ ๆ ในลักษณะของ Web Based Application โปรแกรม Apache เป็นโปรแกรมเว็ปเซิร์ฟเวอร์ที่มีประสิทธิภาพสูง สามารถทำให้เครื่องเซิร์ฟเวอร์ให้การบริการงานเว็ปเซิร์ฟเวอร์ได้อย่างมีประสิทธิภาพ และเป็นโปรแกรมเว็ปเซิร์ฟเวอร์ที่ไม่ต้องเสียค่าใช้จ่าย ซึ่งพร้อมที่จะทำงานทันทีที่ปรับแต่งอีกเพียงเล็กน้อย

หลักการทำงานของ Apache Web Server

Apache Web Server จะเป็นโปรแกรมที่ทำงานเป็น Background Process และคอยรับข้อมูลที่ Port 80 (80 ในที่นี้จะหมายถึง หมายเลขอ้างอิง) ในการสื่อสารข้อมูล ระหว่างโปรแกรม กับโปรแกรม ใน Protocol TCP/IP โดยที่ฝั่งที่ ต้องการข้อมูล เช่น โปรแกรม Browser จะอ้างชื่อ URL (Uniform Resource Locator) เป็นลักษณะ http://www.thailinux.com โดย http:// จะบอกว่าเป็น Hypertext Transfer Protocol ซึ่งเป็นรูปแบบ การรับ-ส่ง ข้อมูลแบบหนึ่ง โดยข้อมูลส่วนใหญ่ ก็จะเป็น text ที่อยู่ในรูปแบบของ HTML (Hypertext Markup Language) และ รูปภาพที่อยู่ในรูปแบบของ .GIF หรือ .JPG โดยจะทำการติดต่อ ไปที่ HTTP Server ที่อยู่ที่ www.thailinux.com ซึ่ง Browser จะไปถาม DNS ที่ติดตั้งในฝั่ง Browser เพื่อแปลงชื่อเป็น IP Address และทำการติดต่อ ไปที่เครื่องที่มี IP Address นี้โดยใช้ Port ปลายทางหมายเลข 80 ซึ่งเป็น Port ที่เป็นมาตรฐานของ HTTP เมื่อ Apache Web Server รับข้อมูลที่ร้องขอเข้ามา ก็จะไปนำข้อมูลของไฟล์ที่ชื่อ index.html ที่อยู่ในไดเร็กทอรีที่ติดตั้งไว้ใน Parameter ชื่อ Documentroot ใน Apache Config ส่งกลับไปให้ Browser และ Browser ก็จะตีความ index.html และแสดงตามคำสั่งต่างๆ ที่อยู่ใน index.html

ส่วนกรณีที่ Click ไปที่ Link ต่างๆ ตัว Browser ก็จะบอกชื่อ File และ ไดเร็กทอรีที่ต้องการไปที่ Web Server และ Web Server ก็จะส่ง ไฟล์ นั้นมาให้ Browser นี่เป็นตัวอย่างง่ายๆ ที่ทำให้เห็นภาพการทำงาน ของ Web Server ในเบื้องต้น

2.2 ลีนุกซ์ คืออะไร

ลีนุกซ์ (Linux) เป็นโปรแกรมระบบปฏิบัติการที่พัฒนาขึ้นโดย Linus Benedict Torvalds โดยมีการพัฒนามาจากระบบปฏิบัติการที่สร้างขึ้นเพื่อใช้งานในการเรียนการสอนวิชาระบบปฏิบัติการ ในมหาวิทยาลัยเฮลซิงกิ ประเทศฟินแลนด์ ที่ชื่อว่า Minix โดยมีลักษณะที่คล้ายกับระบบปฏิบัติการยูนิกซ์ ดังนั้นจึงอาจจะกล่าวได้ว่าลีนุกซ์เป็นระบบปฏิบัติการที่ได้ถอดแบบมาจากระบบปฏิบัติการยูนิกซ์นั่นเอง ซึ่งลีนุกซ์เวอร์ชันแรกที่มีการเผยแพร่ทางระบบอินเตอร์เน็ตนั้นคือเวอร์ชัน 0.02 มีส่วนประกอบเพียงส่วนเคอร์เนล อันถือว่าเป็นหัวใจของระบบปฏิบัติการ มี BASH (Bourne Again Shell) เป็นส่วนติดต่อกับผู้ใช้ สามารถรัน GCC (GNU C Compiler) ได้ เนื่องจากลีนุกซ์เป็นซอฟต์แวร์ที่ทำการเผยแพร่โดยผ่านกลไกของ Free Software Organization หรือที่รู้จักกันในนาม "GNU" ดังนั้นโปรแกรมต้นฉบับของลีนุกซ์จึงได้รับการเผยแพร่ และมีโปรแกรมเมอร์ทั่วโลกนำไปพัฒนาเพื่อให้สามารถทำงานได้บนแพลตฟอร์มที่หลากหลาย แต่อย่างไรก็ตาม ส่วนเคอร์เนลของลีนุกซ์ก็ยังคงอยู่ภายใต้การควบคุมดูแลจากกลุ่มโปรแกรมเมอร์ของ Linus Torvalds อยู่ ทำให้เคอร์เนลของลีนุกซ์มีความเป็นเอกภาพในการที่จะพัฒนา หรือแก้ไขคุณสมบัติต่าง ๆ ที่สำคัญมาก ๆ

ต้นกำเนิด Linux

Linux คือ ระบบปฏิบัติการฟรีแบบ UNIX เป็นระบบปฏิบัติการที่พยายามเลียนแบบ UNIX ผู้ริเริ่มสร้าง Linux คือ Linus Torvalds (โดยได้สร้างในส่วนของ Kernel เท่านั้น) ร่วมกับความช่วยเหลือของนักพัฒนา (Developers) ทั่วโลก Linux สนับสนุน POSIX อิสระ และประกอบด้วย Ture Multitasking , Virtual Memory , Shared Libraries , Demand Loading , Proper memory management , TCP/IP Networking และลักษณะเด่นอื่นๆ ตรงกัน กับ Unix-Type Systems. พัฒนาการ ภายใต้ GNU General Public License

เหตุผลที่เลือกใช้ Red Hat Linux

RedHat เป็นระบบปฏิบัติการที่ได้รับความนิยม ทั้งในประเทศไทย และในต่างประเทศ ซึ่งปัจจุบันได้ออกเวอร์ชันล่าสุด คือ Red Hat 7.2 โดยใช้เคอร์เนล 2.4.7-10 มี Codename ว่า Enigma หากนำมาเปรียบเทียบกับรุ่นก่อน ๆ แล้ว จะสังเกตเห็นได้ว่ามีคุณสมบัติเพิ่มขึ้นหลายอย่างมีเหตุผลมากมายที่ทำให้ RedHat ได้รับความนิยมมากเช่นนี้ ทั้งนี้เนื่องมาจากมีผู้ใช้ลีนุกซ์จำนวนมากที่ใช้

งาน RedHat อยู่ ทำให้มีผู้ที่มีความรู้เกี่ยวกับ RedHat มากกว่าลีนุกซ์ค่ายอื่น เมื่อเกิดปัญหาขึ้นก็สามารถขอคำปรึกษาจากผู้รู้ได้ง่าย นอกจากนี้ยังมีหนังสือ ตำรา เว็ปไซต์ที่มีเนื้อหาอ้างอิงเป็นจำนวนมากกว่าลีนุกซ์ค่ายอื่น ช่วยให้ผู้ที่เริ่มต้นใหม่สามารถค้นคว้าหาข้อมูลได้ด้วยตนเองบางท่านอาจจะเคยได้รับทราบมาว่า ผู้ดูแลระบบเครือข่ายที่ใช้ลีนุกซ์เป็นเซิร์ฟเวอร์อย่างจริงจังมักจะไม่สนใจที่จะใช้ RedHat แต่มักจะเลือกใช้ Distribution อื่น ๆ เช่น Slackware โดยให้เหตุผลว่า RedHat มีการติดตั้งและปรับแต่งระบบที่ง่ายเกินไป ทำให้ผู้ที่ปรับแต่ง และใช้งานมีความรู้ความสามารถน้อยไม่สมกับหน้าที่ผู้ดูแลระบบ แต่อันที่จริงแล้วถ้าหากจะมีระบบปฏิบัติการอะไรซักตัวหนึ่งที่จะประสบความสำเร็จขึ้นจริง ๆ แล้ว ระบบปฏิบัติการนั้นควรจะต้องมีการติดตั้งที่สะดวกรวดเร็ว ปรับแต่งระบบ และค้นหาแก้ไขข้อผิดพลาดได้ง่าย ที่สำคัญจะต้องเป็นระบบปฏิบัติการที่ถูกสร้างขึ้นมาเพื่อผู้ใช้ทุก ๆ ระดับ ตั้งแต่ผู้เริ่มต้น ไปจนถึงมืออาชีพก็สามารถเรียนรู้ และใช้งานได้เหมือนกัน ในขณะเดียวกันก็ยินยอมให้ผู้ใช้งานระดับสูง สามารถแก้ไขปรับแต่งระบบได้โดยวิธีการแบบแมนนวลได้ตามต้องการ ซึ่ง RedHat ก็เป็น Distributionหนึ่งที่มีรูปแบบของการใช้งานเช่นนี้ คือ มีการติดตั้งที่ให้เลือกตั้งแต่แบบง่ายมาก ไปจนถึงแบบขั้นสูง ภายหลังจากการติดตั้งก็สามารถเลือกได้ที่จะใช้การปรับแต่งด้วยโปรแกรมUtilityแบบกราฟฟิก โปรแกรมเมนูแบบเท็กซ์โหมด หรือจะปรับแต่งด้วยโปรแกรมEditorแบบดั้งเดิมก็ได้ตามต้องการในด้านการจัดการเพคเกจ (ซอฟต์แวร์ส่วนประกอบย่อย ๆ ของลีนุกซ์) เป็นอีกสิ่งหนึ่งที่มีความสำคัญมาก ไม่ว่าจะเป็นการติดตั้ง อัพเกรด หรือถอนการติดตั้งซอฟต์แวร์ต่างๆในลีนุกซ์จะต้องมีเครื่องมือที่ช่วยอำนวยความสะดวกสำหรับงานเหล่านี้ RedHat ได้สร้างระบบการจัดการแพคเกจเป็นของตนเองคือ RPM (RedHat Package Management) ซึ่งเป็นที่นิยมใช้งานกันอย่างแพร่หลายดังจะเห็นได้ว่า Distribution อื่น ๆ ก็จะสนับสนุน RPM นี้เช่นกัน รวมทั้งซอฟต์แวร์ใช้งานต่าง ๆ ที่ผลิตขึ้นมาเพื่อติดตั้งใช้งานกับลีนุกซ์ก็จะมีแพคเกจ RPM เป็นส่วนใหญ่ จนแทบจะเรียกได้ว่าเป็นรูปแบบมาตรฐานในการติดตั้งซอฟต์แวร์บนลีนุกซ์ก็ว่าได้ จากความสำเร็จของ RedHat ในด้านความง่ายในการติดตั้งระบบ การปรับแต่งค่า และมีการสนับสนุนทางเทคนิคสำหรับองค์กรขนาดใหญ่ จึงทำให้ RedHat เป็น Distribution อันดับต้น ๆ ที่ถูกเลือกเพื่อนำมาใช้งานในทุก ๆ ระดับ รวมไปถึงเป็นต้นแบบของ Distribution อื่น ๆ ที่ได้รับการพัฒนาขึ้นในเวลาต่อมา จึงทำให้โครงสร้างหลัก ๆ ของลีนุกซ์ค่ายอื่น ๆ ทั้งที่เป็นของคนไทย และของประเทศอื่น ๆ มีความคล้ายกับ RedHat เป็นอย่างมาก ด้วยจำนวนผู้ใช้ RedHat ที่มีจำนวนมากนี้เอง RedHat จึงมีการให้บริการต่าง ๆ เพิ่มขึ้น โดยมีเว็ปไซต์ http://www.redhat.com เป็นแหล่งข่าวสารเกี่ยวกับลีนุกซ์ที่สำคัญแห่งหนึ่ง ซึ่งมีจำนวนสมาชิกนับล้านคนทั่วโลก นอกจากนี้ยังมีสำนักงานสาขาอยู่ทั่วทุกมุมโลก และเมื่อไม่นานมานี้ยังได้ร่วมเป็นพันธมิตรทางธุรกิจ และการพัฒนาผลิตภัณฑ์ร่วมกันกับบริษัทคอมพิวเตอร์รายใหญ่ ๆ เช่น IBM ,Intel ,HP ,Compaq อีกด้วย

RedHat Linux ในฐานะเซิร์ฟเวอร์

ลีนุกซ์ เป็นระบบปฏิบัติการแบบ 32 บิต ที่มีลักษณะคล้ายยูนิกซ์ จึงมีคุณสมบัติที่พร้อมสำหรับการทำหน้าที่เป็นเซิร์ฟเวอร์ของระบบเครือข่ายได้ทันที การนำลีนุกซ์มาใช้งานในระยะแรกจึงเป็นในฐานะเครื่องเซิร์ฟเวอร์มากกว่าจะใช้งานเป็นเครื่องเดสก์ทอป (Desktop) ธรรมดา สำหรับลีนุกซ์ Distribution ต่าง ๆ มักจะออกแบบผลิตภัณฑ์ของตนเองให้ผู้ใช้สามารถนำไปใช้งานได้ทุกลักษณะตามต้องการ จึงได้รวบรวมเอาซอฟต์แวร์ต่าง ๆ เอาไว้ให้เป็นจำนวนมาก RedHat เองก็เช่นกัน เราจึงสามารถนำ RedHat 7.2 มาใช้งานในฐานะเซิร์ฟเวอร์ต่าง ๆ ได้มากมาย

RedHat 7.2 นอกจากจะมาพร้อมกับ Apache 1.3.20 แล้วยังมีส่วนประกอบสำคัญ ๆ ที่ช่วยในการพัฒนาเว็ปอีก เช่น Perl 5.6.0 , Python 1.5.2 , PostgreSQL 7.1.3, MySQL 3.23.41 , PHP 4.0.6 และ Demo โปรแกรมแบบ Commercial อีกจำนวนหนึ่ง

Apache จะถูกรวมเข้าไปกับ Linux หลายเวอร์ชั่นด้วยกันไม่ว่าจะเป็น RedHat, Slackware หรือ Caldera Open Linux สำหรับ RedHat การติดตั้งจะทำผ่าน rpm ทำให้การติดตั้งค่าได้ง่ายมากขึ้น แต่ในกรณีที่ Download จาก Internet จะเป็นรูปแบบของ Tar file และ Zip file การติดตั้งจะยุ่งยากขึ้น แต่สิ่งสำคัญก็คือ การกำหนดค่าให้ Apache ทำงานในแนวทางที่ต้องการหรือการกำหนดเพื่อปรับแต่งค่าในไฟล์ httpd.conf

2.3 UNIX

ระบบปฏิบัติการ (Operating System) คือ กลุ่มของคำสั่งที่ร่วมกันทำงาน เพื่อควบคุมการทำงานของ Hardware และ Software Application อื่นๆ ของคอมพิวเตอร์เราอาจจะแบ่งระบบปฏิบัติการ ตามลักษณะการใช้งานออกเป็น 2 จำพวกคือ

1.Single-User เป็นระบบปฏิบัติการที่ในขณะใดขณะหนึ่งจะให้บริการแก่ผู้ใช้เพียงคนเดียว เป็นระบบปฏิบัติการขนาดเล็กสะดวกในการควบคุมการทำงานเช่น DOS Windows95/98 ฯลฯ

2.Multi-User เป็นระบบปฏิบัติการที่ให้ผู้ใช้มากกว่าหนึ่งคนเข้าทำงานได้พร้อม ๆ กัน โดยการต่อออกเป็น terminal ย่อย ๆ ใช้กับระบบขนาดใหญ่เป็นระบบปฏิบัติการที่ไม่ยึดติดกับระบบเครื่องระบบใดระบบหนึ่ง เป็นระบบปฏิบัติการที่เป็น Multi-user และ Multi-tasking เช่น Unix , Novell , Linux , SunOS ฯลฯ

หน้าที่ของระบบปฏิบัติการที่เป็น Multi User

- I/O คือการนำเข้าและจัดเก็บข้อมูลลงบนอุปกรณ์ของคอมพิวเตอร์ เช่น การบันทึกลง Disk การแสดงผลทางจอภาพ หรือ เครื่องพิมพ์
- การจัดการข้อมูล คือการจัดเก็บข้อมูลเป็นไฟล์ (Files) หรือรวมกันเป็นไดเร็กทอรี

- Command คือคำสั่งที่จะให้ผู้ใช้พิมพ์ให้คอมพิวเตอร์ประมวลผล
- Time Sharing การบริหารเวลาสำหรับการทำงานพร้อมกันหลายๆ งานหรือหลายๆ คน
- โปรแกรมที่ช่วยในการพัฒนาโปรแกรม เช่น Complier_ต่างที่มีอยู่บนระบบปฏิบัติการแต่ละตัว เช่นใน Linux ก็จะมีภาษาต่างๆเช่น C , C++ และอื่นๆอีกหลายภาษา
- ระบบความปลอดภัยของข้อมูลของแต่ละ user ที่คนอื่นไม่สามารถเข้ามากระทำได้โดยมิได้รับอนุญาต
- การติดต่อกันเป็นเครือข่ายเพื่อใช้ทรัพยากรร่วมกัน

ยูนิกซ์ คืออะไร

ยูนิกซ์ (UNIX) เป็นระบบปฏิบัติการ(Operating System) ซึ่งใช้งานบนเครื่องคอมพิวเตอร์ตั้งแต่ ไมโครคอมพิวเตอร์(Micro Computer) จนถึงระดับซูเปอร์คอมพิวเตอร์ (Super Computer)

เริ่มต้นระบบปฏิบัติการยูนิกซ์ (UNIX) ได้ถูกออกแบบโดยห้องปฏิบัติการ AT&T's Bell Lab ในปี ค.ศ.1969 ปัจจุบันยูนิกซ์ได้รับความนิยมมากเนื่องจากสามารถให้บริการผู้ใช้ได้หลายคนในเวลาเดียวกัน (Multiprocessing) โดยที่ผู้ใช้แต่ละคนต่างก็ทำงานได้หลายงานพร้อมๆกัน (Multitasking) อีกด้วยและผู้ใช้สามารถสร้าง เปลี่ยนแปลงแก้ไขคำสั่งต่างๆได้เอง (flexible) ระบบปฏิบัติการยูนิกซ์ มีหลายเวอร์ชันด้วยกัน โดยแบ่งเป็น

- เวอร์ชันต้นแบบจากบริษัท AT&T ซึ่งเรียกว่า System V
- เวอร์ชันที่พัฒนาโดยมหาวิทยาลัยแคลิฟอร์เนียเบิร์กเลย์ ชื่อ BSD
- เวอร์ชันที่ถูกสร้างขึ้นโดยบริษัทผลิตเครื่องคอมพิวเตอร์ต่างๆ เช่น

AIX โดยบริษัท IBM

AUX โดยบริษัท Apple

IRIS โดย บริษัท Silicon Graphic

Linux เป็น Freeware

OSF/I โดย บริษัท DEC

SCO UNIX โดย บริษัท SCO

SunOS โดย บริษัท SUN Microsystem

ULTRIX โดย บริษัท DEC

Basic Command of Unix

- คำสั่ง telnet

เป็นคำสั่งที่เปลี่ยน host ที่ใช้อยู่ไปยัง host อื่น

รูปแบบ :

$ telnet *hostname*

เช่น c:\> telnet student.nu.ac.th เปลี่ยนไปใช้ host ชื่อ student.nu.ac.th

$ telnet 202.44.130.165 เปลี่ยนไปใช้ host ที่มี IP = 202.44.130.165

$ telnet 0 telnet เข้า host ที่ใช้อยู่ในขณะนั้น

เมื่อเข้าไปได้แล้วก็จะต้องใส่ login และ password และเข้าสู่ระบบยูนิกส์นั้นเอง

- คำสั่ง ftp

ftp เป็นคำสั่งที่ใช้ถ่ายโอนไฟล์ข้อมูลจากที่หนึ่ง ไปยังอีกที่หนึ่ง โดยการติดต่อกับ host ที่เป็น ftp นั้นจะต้องมี User Name และมี password ที่สร้างขึ้นไว้แล้ว แต่ก็มี ftp host ที่เป็น public อยู่ไม่น้อยเช่นกัน ดังนั้นจะมี User Name ที่เป็น publicเช่นกัน คือ User ที่ชื่อว่า Anonymous ส่วน password ของ user anonymous นี้จะใช้เป็น E-mail ของผู้ที่จะ connect เข้าไป และโปรแกรมส่วนใหญ่ก็จะอยู่ในไดเร็กทอรีชื่อ pub

รูปแบบ :

$ ftp *hostname*

เช่น c:\windows> ftp bum.nu.ac.th

$ ftp ftp.nectec.or.th

คำสั่ง ftp จะมีคำสั่งย่อยที่สำคัญๆ ได้แก่

ftp> help	ใช้เมื่อต้องการดูคำสั่งที่มีอยู่ในคำสั่ง ftp
ftp> open hostname	ใช้เมื่อต้องการติดต่อไปยัง host ที่ต้องการ
ftp> close	ใช้เมื่อต้องการยกเลิกการติดต่อ ออกจาก host ที่ใช้งานอยู่
ftp> bye หรือ quit	ใช้เมื่อต้องการออกจากคำสั่ง FTP
ftp> ls หรือ dir	ใช้แสดงชื่อไฟล์ที่มีอยู่ใน Current Directory ของ host

ftp> get	ใช้โอนไฟล์ทีละไฟล์จาก host ปลายทางมายัง localhost
ftp> mget	ใช้โอนไฟล์ทีละหลายๆไฟล์จาก host ปลายทางมายัง localhost
ftp> put	ใช้โอนไฟล์ทีละไฟล์จาก localhost ไปเก็บยัง host
ftp> mput	ใช้โอนไฟล์ทีละหลายๆไฟล์จากlocalhost
ftp> cd	ใช้เปลี่ยนไดเร็กทอรี
ftp> delete	ใช้ลบไฟล์

- **คำสั่ง ls**

มีค่าเหมือนกับ คำสั่ง dir ของ dos

รูปแบบ :

```
$ ls [-option] [file]
```

option ที่สำคัญ เช่น

ตารางที่ 2.1 รูปแบบคำสั่ง ls

l	แสดงแบบไฟล์ละบรรทัด แสดง permission , เจ้าของไฟล์ , ชนิด , ขนาด , เวลาที่สร้างไฟล์
a	แสดงไฟล์ที่ซ่อนไว้ (dir /ah)
p	แสดงไฟล์โดยมี / ต่อท้ายไดเร็กทอรี
F	แสดงไฟล์โดยมีสัญลักษณ์ชนิดของไฟล์ต่อท้ายไฟล์คือ / = directory * = execute file @ = link file
ld	แสดงเฉพาะไดเร็กทอรี (dir /ad)
R	แสดงไฟล์ที่อยู่ในไดเร็กทอรีด้วย (dir /s)

เช่น

$ ls

$ ls -la

- **คำสั่ง more**

แสดงข้อมูลทีละหน้าจอ อาจใช้ร่วมกับเครื่องหมาย pipe line (|) หากต้องการดูหน้าถัดไปกด space ดูบรรทัดถัดไปกด Enter

ตัวอย่าง

```
$ ls -la | more
$ more filename
```

- **คำสั่ง cat**

มีค่าเหมือนกับ คำสั่ง type ของ dos ใช้ดูข้อมูลในไฟล์

ตัวอย่าง

```
$ cat filename
```

- **คำสั่ง clear**

มีค่าเหมือนกับ คำสั่ง cls ของ dos ใช้ลบหน้าจอ terminal ให้ว่าง

ตัวอย่าง

```
$ clear
```

- **คำสั่ง date**

ใช้แสดง วันที่ เดือนและปี

ตัวอย่าง

```
$ date 17 May 1999
```

- **คำสั่ง cal**

ใช้แสดง ปฏิทินของระบบ

รูปแบบ $ cal month year

ตัวอย่าง

```
$ cal 07 1999
```

- **คำสั่ง logname**

คำสั่งแสดงชื่อผู้ใช้ขณะใช้งาน

ตัวอย่าง

```
$ logname
```

- **คำสั่ง id**

ใช้แสดงชื่อและกลุ่มของผู้ใช้งาน

ตัวอย่าง

$ id

- **คำสั่ง tty**

แสดงหมายเลข terminal ที่ใช้งานอยู่

ตัวอย่าง

$ tty

- **คำสั่ง hostname**

คำสั่งแสดงชื่อเครื่องที่ใช้อยู่

ตัวอย่าง

$ hostname

- **คำสั่ง uname**

คำสั่งแสดง ชื่อและรุ่นของระบบปฏิบัติการ ชื่อและรุ่นของ cpu ชื่อเครื่อง

ตัวอย่าง

$ uname -a

- **คำสั่ง history**

คำสั่งที่ใช้ดูคำสั่งที่ใช้ไปแล้วก่อนหน้านี้

ตัวอย่าง

$ history เวลาเรียกใช้ต้องมี ! แล้วตามด้วยหมายเลขคำสั่งที่ต้องการ

- **คำสั่ง echo และ banner**

$ echo "Hello" ใช้แสดงข้อความ "Hello" ขนาดปกติ

$ banner "Hello" ใช้แสดงข้อความ "Hello" ขนาดใหญ่

- **คำสั่ง who , w และ finger**

ใช้แสดงว่าใครใช้งานอยู่บ้างขณะนั้น

ตัวอย่าง

$ who

$ w

$ finger ดูผู้ใช้ที่ host เดียวกัน

$ finger @daidy.bu.ac.th ดูผู้ใช้โดยระบุ Host ที่จะดู

$ finger bum ดูผู้ใช้โดยระบุคนที่จะดูลงไป

$ whoami แสดงชื่อผู้ใช้ เวลาที่เข้าใช้งาน และ หมายเลขเครื่อง

$ whoami เหมือนกับคำสั่ง logname

- **คำสั่ง pwd**

แสดงไดเร็กทอรีที่เราอยู่ปัจจุบัน

ตัวอย่าง

$ pwd

- **คำสั่ง mkdir**

ใช้สร้างไดเร็กทอรีเทียบเท่า MD ใน DOS

ตัวอย่าง

$ mkdir dir_name

- **คำสั่ง cp**

ใช้ copy ไฟล์หนึ่งไปยังอีกไฟล์หนึ่ง

รูปแบบ :

```
$ cp [-irfp] file_source file_target
```

option -i หากมีการทับข้อมูลเดิมจะรอถามก่อนที่จะทับ

option -r copy ไฟล์ทั้งหมดรวมทั้งไดเร็กทอรีด้วย

option -f ไม่แสดงข้อความความผิดพลาดออกหน้าจอ

option -p ยืนยันเวลาและความเป็นเจ้าของเดิม

$ cp file_test /tmp/file_test

- **คำสั่ง mv**

ใช้ move หรือเปลี่ยนชื่อไฟล์

รูปแบบ :

```
$ mv [-if] file_source file_target
```

ความหมายของ option เช่นเดียวกับ cp

$ mv index.html main.html เปลี่ยนชื่อไฟล์ index.html เป็น main.html

- **คำสั่ง rm**

ใช้ลบไฟล์หรือไดเร็กทอรีโดยที่ยังมีข้อมูลภายในเทียบเท่า Del และ Deltree ของ Dos

รูปแบบ :

```
$ rm [-irf] filename
```

$ rm [-irf] filename

$ rm -r dir_name ลบ dir_name โดยที่ dir_name เป็นไดเร็กทอรีว่างหรือไม่ว่างก็ได้

$ rm -i * ลบทุกไฟล์โดยรอการยืนยัน

- **คำสั่ง rmdir**

ใช้ลบไดเร็กทอรีที่ว่าง เทียบเท่ากับ rd ของ Dos

รูปแบบ :

```
$ rmdir dir_name
```

- **คำสั่ง alias**

ใช้ย่อคำสั่งให้สั้นลง

$ alias l = ls -l

$ alias c = clear

- **คำสั่ง unalias**

ใช้ยกเลิก alias เช่น

$ unalias c

- **คำสั่ง type**

ใช้ตรวจสอบว่าคำสั่งที่ใช้เก็บอยู่ที่ใดของระบบ

รูปแบบ :

```
$ type command
$ type clear
```

- **คำสั่ง find**

ใช้ค้นหาไฟล์ที่ต้องการ เช่น

$ find /usr/bin -name "*sh" -print : หาไฟล์ที่ลงท้ายด้วย sh จาก /usr/bin

- **คำสั่ง grep**

ใช้ค้นหาข้อความที่ต้องการจากไฟล์

$ grep ข้อความ file

- **คำสั่ง man**

man เป็นคำสั่งที่เป็นคู่มือการใช้คำสั่งแต่ละคำสั่งเช่น

$ man ls

$ man cp

- **คำสั่ง write**

ใช้ส่งข้อความไปปรากฏที่หน้าจอของเครื่องที่ระบุในคำสั่งไม่สามารถใช้ข้าม host ได้
เช่น

$ write s0460003

- **คำสั่ง mesg**

$ mesg : ดู status การรับการติดต่อของ terminal

$ mesg y : เปิดให้ terminal สามารถรับการติดต่อได้

mesg n : ปิดไม่ให้ terminal สามารถรับการติดต่อได้

- **คำสั่ง talk**

ใช้ติดต่อสื่อสารแบบสองทาง เหมือนกับการคุยโดยผู้ส่ง ๆ ไปแล้วรอการตอบกลับจาก ผู้รับ
สามารถหยุดการติดต่อโดย Ctrl + c สามารถใช้ข้าม host ได้
รูปแบบ :

```
$ talk username@hostname
```

- **คำสั่ง pine**

ใช้อ่านและส่งจดหมายข้างในจะมี menu ให้ใช้

- **คำสั่ง tar**

ใช้สำหรับ รวมไฟล์ย่อยให้เป็นไฟล์ Packet คล้ายๆกับการ zip หลายๆไฟล์ให้เป็นไฟล์เดียวแต่ขนาดไฟล์ไม่ได้ลดลงอย่างการ Zip โดยไฟล์ Output ที่ได้จะตั้งชื่อเป็น filename.tar หรือการแตกไฟล์ packet จาก filename.tar ให้เป็นไฟล์ย่อยมักจะใช้คู่กับ gzip หรือ compress เพื่อทำการลดขนาด packet ให้เล็กลง

รูปแบบ :

```
$ tar -option output input
```

-option ประกอบด้วย -cvf , -tvf , -xvf แสดงด้านล่าง

output คือ ไฟล์.tar หรืออาจจะเป็น device เช่น tape ก็ได้

input คือ ไฟล์หรือกลุ่มไฟล์หรือไดเร็กทอรีหรือรวมกันทั้งหมดที่กล่าวมา

$ tar -cvf Output_file.tar /home/myhome/*

Option -cvf ใช้สำหรับการรวมไฟล์ย่อยไปสู่ไฟล์ .tar จากตัวอย่าง รวมไฟล์ทุกไฟล์ที่อยู่ใน /home/myhome/ เข้าสู่ไฟล์ชื่อ Output_file.tar

$ tar -tvf filename.tar

Option -tvf ใช้แตกไฟล์ .tar เป็นไฟล์ย่อยๆแบบ preview คือแสดงให้ดูไม่ได้แตกจริงอาจใช้คู่กับ คำสั่งอื่น เพื่อให้ได้ประโยชน์ตามต้องการ เช่น tar -tvf filename.tar |more

$ tar -xvf filename.tar

Option -xvf ใช้แตกไฟล์ .tar เป็นไฟล์ย่อยๆ โดยจะแตกลงใน Current Directory

- **คำสั่ง gzip**

ใช้ zip หรือ Unzip ไฟล์ packet โดยมากแล้วจะเป็น .tar เช่น

$ gzip filename.tar : ผลที่ได้จะได้ไฟล์ซึ่งมีการ zip แล้วชื่อ filename.tar.gz

$ gzip -d filename.tar.gz : ใช้ unzip ไฟล์ผลที่ได้จะเป็น filename.tar

- **คำสั่ง Compress และ Uncompress**

หลังจากการ compress แล้วจะได้เป็นชื่อไฟล์เดิมแต่ต่อท้ายด้วย .Z การใช้งานเหมือนกับ gzip และ gzip -d เช่น

$ compress -v file.tar : จะได้ไฟล์ชื่อ file.tar.Z โดย Option -v จะเป็นการ verify การ compress

$ uncompress -v file.tar.Z

Operating System Component

1. Kernel คือหัวใจของระบบจะควบคุมการทำงานภายในทั้งหมดของระบบคอมพิวเตอร์ เช่น การเตรียมทรัพยากรต่างๆของระบบ การจัดเก็บข้อมูล การบริหารหน่วยความจำ การควมคุมอุปกรณ์ต่างๆที่ต่ออยู่ ตัว Kernel จะขึ้นกับชนิดของเครื่องดังนั้นเราต้องใช้ Kernel คนละตัวกันหากใช้เครื่องคนละตระกูลกัน

2. File System (FS) คือโครงสร้างการจัดเก็บข้อมูลในฮาร์ดดิสก์ เพื่อให้ระบบปฏิบัติการสามารถอ่านเขียนใช้ไฟล์ที่ต้องการได้อย่างมีประสิทธิภาพ โดยที่ระบบปฏิบัติการแต่ละตัวจะมี FS ที่แตกต่างกัน เช่น

ตารางที่ 2.2 ระบบไฟล์ของแต่ละระบบปฏิบัติการ

Operating System	**File System**
DOS/Windows95	FAT12,FAT16
Windows98/95-osr	FAT12,FAT16,FAT32
Windows NT	NTFS,FAT16,HPFS
OS/2	FAT12,FAT16,HPFS
Linux	EXT2,VFAT,HPFS,NTFS,etc.
SunOS	UFS
ฯลฯ	ฯลฯ

หมายเหตุ เนื่องจาก Linux ใช้ File System แบบ Ext2 (Extended Files System 2) จึงทำให้ Linux สามารถมองเห็นดิสก์ก้อนที่ใหญ่มากมีขนาดถึง 4 เทราไบต์(Tbytes) หรือขนาด 4000 Gbytes

3. Shell เป็น Command Interpreter เป็นตัวกลางติดต่อระหว่าง User กับ Kernel คอยรับคำสั่งที่จะพิมพ์เข้าไปแล้วแปลคำสั่งนั้นต่อไป นอกจากนี้แล้วยังสามารถที่จะนำเอาคำสั่งต่างๆ มาเขียนเป็นโปรแกรมเรียกว่า Shell Script และ Shell ยังสามารถกำหนดทิศทาง Input / Output ได้ด้วยการเปลี่ยนทิศทางจะมีเครื่องหมายที่จำเป็นคือ

"**>**" หมายถึง การเปลี่ยนทิศทางของ output

"**<**" หมายถึง การเปลี่ยนทิศทางของ input

"**>>**" หมายถึง การเปลี่ยนทิศทางของ output ไปต่อท้ายไฟล์

การทำงานผ่าน Shell มี 2 ลักษณะคือ

- **Synchronous Execution** เป็นการทำงานตามลำดับของคำสั่งทีละคำสั่งจนเสร็จแล้วจึงจะขึ้น prompt เพื่อป้อนคำสั่งต่อไป เรียกว่าการทำงานแบบฉากหน้า (foreground mode) เช่น

  $ ls -l (เป็นการ list ดูไฟล์แบบยาวในไดเร็กทอรีปัจจุบัน)

- **Asynchronous Execution** จะทำงานตามคำสั่งโดยที่งานเก่าจะเสร็จแล้วหรือยังไม่เสร็จก็ตามแต่ Shell จะกำหนด prompt และสร้าง Shell ใหม่ขึ้นมาเพื่อรองรับงานใหม่ต่อไป เรียกว่าการทำงานแบบฉากหลัง (Background mode) การทำงานแบบนี้ทำได้โดยเติมเครื่องหมาย Ampersand (&) ไว้ที่ท้ายคำสั่งนั้นเช่น

  $ netscape & (เรียกโปรแกรม netscape แล้วขึ้น prompt โดยไม่ต้องรอให้ออกจาก netscape ก่อน)

Shell ที่นิยมใช้

- **Bourne Shell** (sh) เป็น Starndard Shell ที่มีใน Unix ทุกตัวสามารถย้าย Shell Script ไปยัง Unix ระบบอื่นได้โครงสร้างเป็นแบบ Algol สามารถใช้งาน Procedure ได้ จะมี Default prompt เป็น "$"
- **C Shell** (csh) มีโครงสร้างคล้ายภาษา C ทำงานได้ดีกว่า Bourne Shell มีไฟล์ที่เก็บคำสั่งที่ใช้ไปแล้ว ทำงานกับ Shell Script ของ Bourne Shell ไม่ได้ Default prompt เป็น "%"
- **Korn Shell** (ksh) ทำงานได้ดีกว่า sh และ csh แต่ไม่ได้มีใน Unix ทุกตัว ksh มีขนาดใหญ่กว่า Shell อื่น ๆ เขียน Shell Script ได้ง่ายขึ้นและรัดกุม เป็น Standard IEEE PDSIX 1003.2 default prompt เป็น "$"
- **Bourne Again Shell** (bash) เป็นการพัฒนา sh ให้สามารถมีแฟ้มคำสั่งที่ใช้ไปแล้ว และเพิ่มขีดความสามารถเพิ่มขึ้นอีกหลายอย่าง (default of Linux) default prompt เป็น "$"

4.Utilities คำสั่งต่างที่ทำงานได้บน ระบบงาน Unix จึงทำให้ kernel มีขนาดเล็ก เพราะจะมีเฉพาะหน้าที่สำคัญเท่านั้น

ประเภทของไฟล์ใน UNIX

ไฟล์ในระบบยูนิกซ์นั้นจะขึ้นอยู่กับผู้สร้างยูนิกซ์แต่ละตัวซึ่งมีทั้งแตกต่างและเหมือนกัน และการตั้งชื่อไฟล์ในระบบยูนิกซ์ส่วนใหญ่จะสามารถตั้งชื่อได้ยาวถึง 255 ตัวอักษรโดยที่ตัวอักษรตัวเล็ก และตัวอักษรตัวใหญ่นั้นมีความแตกต่างกัน สามารถใช้ตัวเลขหรือขีดเส้นใต้ร่วมด้วยก็ได้ แต่ไม่ควรใช้เครื่องหมายเหล่านี้มาตั้งชื่อ เช่น ^ " ' , - ?] () ~ ! $ @ # <> $ / และหากไฟล์ใดที่ตั้งชื่อ

ขึ้นต้นด้วยจุด "." จะทำให้ไฟล์นั้นเป็น hidden file คือไฟล์ที่ถูกซ่อนไว้ จะไม่สามารถมองเห็นได้โดยใช้คำสั่งทั่วไปจะต้องมี option เพิ่มเติม

- **Regular files** คือไฟล์ทั่วไปที่สร้างขึ้นได้ด้วย Text Editor หรืออาจจะสำเนามาจากไฟล์อื่น หรืออาจจะเป็นโปรแกรมใช้งานต่างๆก็ได้
- **Directory files** คือไฟล์ที่เก็บไฟล์ทั่วไปหรือจะเก็บไฟล์ที่เป็นไดเร็กทอรีด้วยกัน ที่เรียกว่า Sub Directory ก็ได้ โดยที่ไดเร็กทอรีท้ายสุด (Root) ของ ยูนิกซ์จะแทนด้วย " / "
- **Special files** เป็นไฟล์พิเศษจะมีอยู่สองแบบคือ Character device file และ Block device file ทั้งสองแบบจะเป็นไฟล์ Device Driverโดยส่วนใหญ่จะเก็บไว้ที่ /dev แต่ไฟล์ทั้งสองจะแตกต่างกัน ที่การรับส่งข้อมูล นั่นคือ Character device file จะรับส่งข้อมูลที่ละตัวอักษร แต่ Block device file จะรับส่งข้อมูลเป็นบล็อก
- **Unix Domain Sockets** ใน BSD Unix หรือ Name pipes ใน AT&T Unix
- **Symbolic Link files** หรือไฟล์เชื่อมต่อ การเชื่อมต่อของไฟล์มี 2 ลักษณะคือ

1. **Hard Link** การเชื่อมต่อแบบนี้จะใช้ I-node เดียวกับไฟล์ต้นฉบับ เหมือนกับมีการสร้างไฟล์ใหม่ แต่ใช้ค่า I-node เดิม และ I-node จะมีตัวนับจำนวนไฟล์ที่เชื่อมต่อด้วย หากแก้ไขไฟล์ใดไฟล์หนึ่งจะมีผลกระทบส่งถึงกัน เพราะข้อมูลเก็บที่เดียวกัน แต่ข้อมูลต้องอยู่ที่ partition เดียวกัน ทำให้ประหยัดเนื้อที่ สามารถอ้างถึงข้อมูลได้จากหลายๆที่

2. **Symbolic Link** การเชื่อมต่อแบบนี้จะสร้าง I-node ของตัวเองขึ้นมาใหม่ เหมือนกับ shortcut ของ Windows 95 โดยที่หากเปลี่ยนแปลงต้นฉบับจะมีผลกับ link file แต่หากลบ link file จะไม่มีผลใดๆต่อไฟล์ต้นฉบับ สามารถใช้ได้ทั้งที่อยู่ partition เดียวกัน หรือต่าง partition กันก็ได้ เราสามารถที่จะแยกประเภทของไฟล์ต่างได้โดยใช้คำสั่ง ls -l แล้วจะแสดงสัญลักษณ์ โดยจะแสดงดังนี้

ตารางที่ 2.3 แยกประเภทของไฟล์

Type	Symbol	Create	Remove
Text file	-	cp , mv ,etc	rm
Directory	p	mkdir	rm -r , rmdir
Character device	v	mknod	rm
Block device	b	mknod	rm
Unix domain socket	s	socket	rm
Name pipes	p	mknod	rm
link file	l	ln -s	rm

โครงสร้างไฟล์ไดเร็กทอรีของระบบยูนิกซ์ส่วนใหญ่จะเป็นแบบ File system Hierarchy Standard (FHS) โดยการจัดลำดับชั้นจะเป็นแบบต้นไม้หัวกลับ โดยเริ่มจากชั้นแรกที่เป็น ราก หรือ Root เขียนแทนด้วย / ไฟล์แต่ละไฟล์อาจจะสร้างขึ้นมาเองหรือเป็นโปรแกรมก็ได้ ไฟล์ลักษณะนี้จะเป็นไฟล์ไดเร็กทอรี การจัดไฟล์ระบบนี้จะทำให้การจัดไฟล์เป็นระบบและง่ายต่อการดูแลรักษา โดยจะมีโครงสร้างหลักเป็นดังนี้

- / : เป็นไดเร็กทอรี Root ที่เก็บไฟล์ Kernel ของระบบ
- /bin : เป็นไดเร็กทอรีที่ใช้เก็บคำสั่งทั่วไปของระบบ
- /dev : เป็นไดเร็กทอรีที่ใช้เก็บไฟล์ที่เกี่ยวกับอุปกรณ์ต่างๆ
- /etc : เป็นไดเร็กทอรีที่ใช้เก็บไฟล์ที่เป็น config files ของเครื่อง
- /etc/X11 : เป็นไดเร็กทอรีที่ใช้เก็บไฟล์ที่เป็น config files ของ x - windows
- /etc/skel : เป็นไดเร็กทอรีที่ใช้เก็บไฟล์ที่เป็นไฟล์ต้นฉบับที่จะถูกสำเนาไปยัง Home User
- /lib : เป็นไดเร็กทอรีที่ใช้เก็บไฟล์ไลบรารี สำหรับให้โปรแกรมต่างๆเรียกใช้
- /sbin : เป็นไดเร็กทอรีที่ใช้เก็บไฟล์คำสั่งของผู้ดูแลระบบ
- /usr : เป็นไดเร็กทอรีที่ใช้เก็บไฟล์โปรแกรมของผู้ใช้ทั่วไป
- /var : เป็นไดเร็กทอรี ที่ใช้เก็บไฟล์ข้อมูลทั่วไปของระบบ

PERMISSION

ยูนิกซ์เป็นระบบปฏิบัติการที่ใช้ไฟล์ต่างๆ ร่วมกันหากทุกคน มีสิทธิที่จะกระทำต่อทุกไฟล์เท่ากัน ย่อมจะทำให้เกิดความวุ่นวาย ดังนั้นในระบบยูนิกซ์จึงมี User ID และ Group ID ประจำ User แต่ละคน จึงทำให้ที่ Home Directory ของแต่ละ User จะเป็นที่ ที่ User แต่ละคนมีสิทธิมากที่สุด เมื่อ User สร้างไฟล์ขึ้นมาก็จะทำให้ มีชื่อของผู้สร้างติดอยู่ด้วย การจำกัดสิทธิการเข้าถึงไฟล์ออกเป็น 3 กลุ่มคือ

Owner : เจ้าของไฟล์หรือผู้ที่สร้างไฟล์

Group : ผู้ใช้กลุ่มเดียวกับผู้ใช้ไฟล์ คือ ผู้ใช้ที่มี gid เดียวกับเจ้าของไฟล์

Other : คนอื่นๆหรือใครก็ได้

สิทธิในไฟล์จะประกอบไปด้วย

Read Permission สิทธิในการอ่าน แทนด้วย r

Write Permission สิทธิในการเขียน แทนด้วย w

Execute Permission สิทธิในการ Run แทนด้วย x

user สามารถที่จะดู Permission ของไฟล์และชนิดของไฟล์ได้โดยคำสั่ง

$ ls -la

-rwxr--r—1 bum Special 5223 May 12 10:10 .profile

-rwxr--r-- 1 bum Special 2022 May 12 10:13 .kshrc

drwx------ 2 bum Special 1024 May 12 10:34 mail

-rw-r--r-- 1 bum Special 11211 May 12 11:01 test

จากตัวอย่างจะเห็นว่า มีทั้งหมด 7 filed ดังนี้

ตารางที่ 2.4 ความหมายของฟิลด์

Field	ความหมาย
1	File Type และ Permission
2	จำนวน link
3	เจ้าของ (owner)
4	กลุ่ม (group)
5	ขนาดของไฟล์ (byte)
6	วัน-เวลาที่ update
7	ชื่อไฟล์

ดูที่ field ที่ 1 ที่เป็น Permission โดย

อักษรตัวที่ 1 แสดงชนิดของไฟล์

อักษรตัวที่ 2-4 แสดง Owner

อักษรตัวที่ 5-7 แสดง Group

อักษรตัวที่ 8-10 แสดง Other

เช่นจากตัวอย่าง ไฟล์ .kshrc มี permission เป็น -rwxr--r-- หมายความว่า Owner สามารถที่จะ อ่าน เขียน และ Run ได้ แต่ user กลุ่มเดียวกับ owner และ other อ่านได้เพียงอย่างเดียว สังเกตุได้ว่าหากไม่มี permission จะแสดงด้วย

คำสั่งเปลี่ยน Permission

การเปลี่ยน permission ของไฟล์กระทำได้โดยผู้ที่เป็น Admin ของระบบ หรือเจ้าของไฟล์นั้น โดยมีคำสั่งคือ

1.คำสั่ง chmod ใช้เปลียน permission ของไฟล์มีวิธีการเปลี่ยนได้ 2 วิธี คือ

- **Absolute Permission**

  รูปแบบ $ chmod ตัวเลข filename

  โดยสามารถหาตัวเลขที่มาใส่ได้จากการแทนค่าน้ำหนักของแต่ละบิทลงไปคือ

  บิท r แทนน้ำหนักด้วย 4

  บิท w แทนน้ำหนักด้วย 2

  บิท x แทนน้ำหนักด้วย 1

  บิท - แทนน้ำหนักด้วย 0

  โดยหากต้องการให้ permission ใดก็แทนค่าของบิทนั้นลงไปแล้วนำเลขน้ำหนักของแต่ละบิทมารวมกัน (คิดทีละส่วนโดยแยกเป็น Owner , Group และ Other) เช่น จะกำหนดสิทธิไฟล์ test ให้ Owner สามารถอ่าน เขียน และทำงานได้ Group สามารถอ่าน และทำงานได้ ส่วน Other สามารถทำงานได้เพียงอย่างเดียว โดยคำนวนได้ดังนี้

  Permission rwx r-x --x

  Number 7 5 1

  ใช้คำสั่ง :

  ```
  $ chmod 751 test
  ```

- **Relative Permission**

ตารางที่ 2.5 สิทธิ์ในแต่ละผู้ใช้

ผู้ใช้ไฟล์	เครื่องหมาย	สิทธิ
u (เจ้าของไฟล์)	+ เพิ่มสิทธิ	r (อ่าน)
g (กลุ่มเดียวกับเจ้าของไฟล์)		
o (คนทั่วไปใครก็ได้)	- ลดสิทธิ	w (เขียน)
a (ทุกคนทุกกลุ่มที่กล่าวมา)	= กำหนดสิทธิ	x (Run)

เช่นจะเปลี่ยน permission ของไฟล์ .kshrc จาก rwxr--r-- เป็น rwxrw-r--

$ chmod g+w .kshrc

หรือจะเปลี่ยน permission ของไฟล์ .profile จาก rwxr--r-- เป็น rwxrw-rw-

$ chmod go+w .profile

2. คำสั่ง chown ใช้เปลี่ยนผู้เป็นเจ้าของไฟล์เดิม เช่น

$ chown newuser test คือเปลี่ยน field ที่ 3 จากการใช้คำสั่ง ls -la จากเจ้าของเดิมคือ bum เป็น newuser

3. คำสั่ง chgrp ใช้เปลี่ยนกลุ่มผู้เป็นเจ้าของไฟล์ เช่น

$ chgrp newgroup test คือเปลี่ยน field ที่ 4 จากการใช้คำสั่ง ls -la จากเจ้าของเดิมคือ Special เป็น newgroup

Text Editor

Text Editor ที่ใช้ในระบบยูนิกซ์ที่เห็นบ่อยคือ โปรแกรม pico และโปรแกรม vi แต่ pico ไม่ได้มีอยู่ใน Unix ทุกตัว การใช้งานง่าย ไม่ต้องจำคำสั่งต่างเพราะมีอธิบายอยู่แล้วที่ด้านล่างหน้าจอภาพ สามารถพิมพ์ Text ได้เลย แต่ Text Editor ที่ชื่อ vi จะเป็น Text Editor ที่มีอยู่ในทุกยูนิกซ์ การใช้งานค่อนข้างยาก ดังนั้นผู้เขียนจะแนะนำเฉพาะการใช้ vi เท่านั้น

การเรียกใช้งาน Text Editor

$ pico filename หรือ $ pico

$ vi filename หรือ $ vi

การใช้งาน vi

vi เป็น Text Editor ที่มีบนยูนิกซ์ จะแบ่งการทำงานออกเป็น 3 mode คือ

1. **Command Mode** เป็นการทำงานของการเคลื่อนย้าย cursor (Editor ตัวอื่นจะใช้คีย์ลูกศร ,Home ,End ,insert , delete แต่ใน vi คีย์เหล่านี้จะไม่มีผล)
2. **Edit Mode** เป็นการทำงานของการแก้ไขข้อความ
3. **Last Line Mode** เป็นการ save , open , quit , ค้นหา , ฯลฯ
   การเปลี่ยน mode ใน vi จะใช้ปุ่ม Esc ยกเว้นเปลี่ยนไปสู่ Last line Mode จะต้องกด Esc แล้วกด Shift + : จะปรากฏ : ที่บรรทัดล่างสุด

- **Command Mode**

การทำงานใน mode นี้จะเป็นการเคลื่อนย้ายเคอร์เซอร์ไปยังตำแหน่งที่ต้องการ แต่หากย้ายไปตำแหน่งที่ไม่มีข้อมูล มันจะส่งเสียงเตือน ตัวอักษรที่ใช้ใน mode นี้ที่สำคัญได้แก่

ตารางที่ 2.6 ตัวอักษรที่ใช้ใน Command Mode

h	เลื่อน cursor ไปทางซ้ายทีละตัวอักษร
J	เลื่อน cursor ลง 1 บรรทัด
K	เลื่อน cursor ขึ้น 1 บรรทัด
L	cursor ไปทางขวาทีละตัวอักษร
W	เลื่อน cursor ไปทางขวาทีละคำ
B	เลื่อน cursor ไปทางซ้ายทีละคำ
$	เลื่อน cursor ไปท้ายบรรทัด
0	เลื่อน cursor ไปต้นบรรทัด
NG	ไปยังบรรทัดที่ n หากไม่ใส่ n จะไปบรรทัดสุดท้าย
Ctrl+f	เลื่อน cursor ลง 1 หน้าจอ
Ctrl+b	เลื่อน cursor ขึ้น 1 หน้าจอ

ตารางที่ 2.6 (ต่อ)

Ctrl+L	Refresh หน้าจอ
[[	ไปยังต้นไฟล์
]]	ไปยังท้ายไฟล์
yy	Copy ข้อความทั้งบรรทัด
yw	Copy ข้อความทั้งคำ
yG	Copy ถึงท้ายไฟล์
y$	Copy ถึงท้ายบรรทัด
P	Paste หลัง cursor
P	Paste หน้า cursor
Cw	พิมพ์ทับทีละ word
c$	พิมพ์ทับจนถึงท้ายบรรทัด
CG	พิมพ์ทับจนถึงท้ายไฟล์
R	พิมพ์ทับทีละ 1 ตัว
R	พิมพ์ทับจนกว่าจะกด Esc
U	Undo การกระทำครั้งล่าสุด
X	ลบตรง cursor
X	ลบหน้า cursor
Dw	ลบคำ
Dd	ลบบรรทัด
d$	ลบจาก cursor จนท้ายบรรทัด
d0	ลบจาก cursor จนต้นบรรทัด
DG	ลบจาก cursor จนท้ายไฟล์

- **Edit Mode**

ตัวอักษรที่ใช้ใน mode นี้ที่สำคัญได้แก่

ตารางที่ 2.7 ตัวอักษรที่ใช้ใน Edit Mode

a	เพิ่มข้อมูลต่อจาก cursor
A	เพิ่มข้อมูลต่อจากท้ายบรรทัด
i	เพิ่มข้อมูลหน้า cursor
I	เพิ่มข้อมูลที่ต้นบรรทัด
o	แทรกบรรทัดด้านล่าง cursor
O	แทรกบรรทัดด้านบน cursor

- **Last Line Mode**

การใช้งาน mode นี้ก็กด "Esc"" แล้วกด ":" ก็จะปรากฏ ":" ที่ท้ายบรรทัด และสามารถที่จะป้อนคำสั่งต่อไปนี้ได้

ตารางที่ 2.8 ตัวอักษรที่ใช้ใน Last Line Mode

:q!	quit
:w!	save
:wq!	save and quit
:w! filename	save as filename
:e! filename	open filename
:/string	ค้นหาข้อความที่ต้องการ
:help	ดูคำสั่งต่างๆ
:set nu	แสดงหมายเลขบรรทัด
:set nonu	ไม่แสดงหมายเลขบรรทัด

Processes

โปรเซส คือ โปรแกรมที่กำลังทำงานอยู่ขณะนั้น เพราะในระบบยูนิกซ์จะเป็นแบบ Multi Programming ทำให้ ไมโครโปรเซสเซอร์(CPU) สลับเวลากันทำงานโดยเอาเวลาที่รอการติดต่อการของอุปกรณ์ไปใช้ทำงานอื่น แต่มันทำด้วยความเร็ว เราจึงคิดว่ามันทำงานขนานกัน แต่จริงๆ แล้ว CPU จะทำทีละงาน โปรเซสที่ถูกสร้างใหม่จะเป็นโปรเซสลูก (Child process) ของโปรเซสเดิมหรือโปรเซสแม่ (parent process) โดยโปรเซสทั้งสองต่างกันที่ Process Id (Process Identification) หรือ PID โดยที่ PPID (Parent Process Identification) ของโปรเซสลูกจะเหมือนกับ PID ของโปรเซสแม่ที่สร้างมันขึ้นมา โดย Kernel จะมีส่วนที่ตัดสินใจ (Scheduling) ว่าจะเลือกทำโปรเซสใดที่สำคัญกว่าก่อน ซึ่งขึ้นอยู่กับเจ้าของโปรเซสนั้น

ชนิดของโปรเซส

1. **Foreground Process** เป็นโปรเซสทั่วไปที่ทำงานปกติ คือรับคำสั่งแล้วทำงานจนเสร็จ แล้วขึ้น prompt รอรับคำสั่งใหม่
2. **Background Process** เป็นโปรเซสที่สำคัญน้อยกว่า foreground process มักจะเป็นโปรเซสที่ใช้เวลาทำงานนาน ไม่ต้องติดต่อผู้ใช้เช่นการคอมไพล์โปรแกรม สร้างได้โดยเรียกใช้คำสั่งแล้วใส่เครื่องหมาย & ต่อท้าย
3. **Daemon Process** เป็น Background Process ที่ทำงานตั้งแต่เริ่มต้นระบบจนกว่าจะยกเลิกระบบ จะใช้งานเมื่อถูกเรียกจาก I/O โปรเซสนี้ชื่อโปรเซสจะลงท้ายด้วยตัว d
4. **Zombie Process** เป็นโปรเซสที่ค้าง ไม่ทำงานแล้ว แต่ก็ไม่สามารถหลุดได้ด้วยตัวเอง

การดูและกำจัดโปรเซส

คำสั่งสำคัญที่เกี่ยวข้องคือ ps , killl , grep

- ps คือคำสั่งที่ใช้ดูสถานะของโปรเซส
- kill คำสั่งที่ใช้กำจัดโปรเซส
  รูปแบบ $ kill -[เลขสัญญาณ] PID
- grep ใช้ร่วมกับ ps เพื่อช่วยค้นหาโปรเซสที่ต้องการ

$ ps -ef | gerp bum เป็นการขอดูโปรเซสทั้งหมดแบบเต็มรูปแบบโดยเลือกเอาเฉพาะโปรเซสที่มีข้อความว่า bum

ตารางที่ 2.9 การดูและกำจัดโปรเซส

UID	PID	PPID	TTY	TIME	COMD
bum	20023	20035	pts/13	0:05	ksh
bum	20032	20022	pts/13	0:01	who

UID คือ เจ้าของโปรเซส

PID คือ process id

PPID คือ parent process id

TTY คือ terminal ที่โปรเซสทำงานอยู่

TIME คือ เวลาสะสมโปรเซสใช้

COMD คือ ทำสั่งที่ทำให้เกิดโปรเซส

$ kill -9 20032 เป็นการกำจัดโปรเซสที่มี PID = 20032

ข้อควรจำ หากกำจัดโปรเซสแม่จะทำให้โปรเซสลูกกลายเป็น zombie process

การเข้า - ออกระบบยูนิกซ์ (Login - exit)

เมื่อระบบยูนิกส์พร้อมที่จะใช้งานที่หน้าจอจะปรากฏ ข้อความรอให้ผู้ใช้ใส่

login : _

นั่นคือผู้ใช้จะต้องใส่ User Name หรือ Account ซึ่ง Admin ของระบบจะเป็นผู้ออกให้ ซึ่งการใส่ Login Name นี้ต้องคำนึงถึงตัวอักษรตัวเล็กตัวใหญ่ด้วยแต่โดยมากแล้วจะเป็นอักษรตัวเล็กทั้งหมด เช่น

login :*bum*

เมื่อ Login Name แล้วระบบจะให้เราใส่ password ซึ่งก็เช่นกันต้องคำนึงถึงตัวอักษรตัวเล็กตัวใหญ่ด้วย ซึ่งการ Login ครั้งแรกอาจจะยังไม่มี passwd หรือ อาจมีมาแล้วพร้อมกับ Login Name ขึ้นอยู่กับ Admin ของระบบที่ใช้ออยู่ โดยหากไม่มี password ระบบจะบังคับให้เราใส่ password สองครั้งเพื่อเป็นการยืนยันโดยข้อความที่พิมพ์ไปจะไม่มีการแสดงใดๆเกิดขึ้นโดย password จะต้องยาวอย่างน้อย 6 ตัวอักษรและต้องมีตัวเลขอย่างน้อยสองตัว

New Password :*ใส่ password ที่ต้องการ*

Retry Password :*ใส่ password ที่ต้องการอีกครั้ง*

หากใส่ไม่ตรงกันเครื่องจะให้ใส่ใหม่อีกครั้งจนกระทั้งใส่ตรงกันทั้งสองครั้ง แต่หากว่าระบบที่ใช้ถูกตั้ง password มาแล้วก็ใส่ตามปกติเช่น

login : *bum*

passwd :

แต่หากต้องการเปลี่ยน password เก่าก็สามารถทำได้โดย

$ *passwd*

Old Password :ใส่ password เก่า

New Password :ใส่ password ใหม่ที่ต้องการ

Retry Password :ใส่ password ใหม่ที่ต้องการอีกครั้ง

พิจารณาเพิ่มเติมถึงการ Login ของแต่ละ Shell ว่ามีการอ่าน Config ไฟล์ใดบ้างเมื่อทำการ Login ของแต่ละ User โดยการปรับแต่งค่าไฟล์เหล่านี้ก็เปรียบได้กับ autoexec.bat ใน Dos หรือ Windows 95 ว่าจะให้ทำคำสั่งอะไรบ้างเมื่อตอน ที่เริ่มเข้าสู่ระบบ โดยรูปแบบการเขียน Config ไฟล์นี้จะเขียนแบบ Shell Script หรือบางครั้งเราจะเรียกไฟล์นี้ว่า Script File ลำดับการอ่านของแต่ละ Shell เป็นดังนี้

- ksh (korn shell) จะอ่านตามลำดับคือ /etc/profile ==> ~/.profile ==> ~/.kshrc (~/ คือ home directory ของแต่ละ user)
- csh (C shell) จะอ่านตามลำดับคือ /etc/profile ==> ~/.login ==> .cshrc
- sh (Bourne shell) จะอ่านตามลำดับคือ /etc/profile ==> ~/.profile

ทดลองใช้คำสั่งต่าง จนกระทั้งคิดว่าเสร็จแล้วเราไม่ควรที่จะปิดเครื่องโดยที่ยังไม่ได้ Log Out การออกจากระบบทำได้โดย

1. $ *exit*
2. $ *logout*
3. $ *^d* (Ctrl + d)

หมายเหตุ การ กด Ctrl + c จะเป็นการหยุดการทำงานของงานที่เราทำอยู่ในขณะนั้น

2.4 Unix System Administration เบื้องต้น

1. Super User

Super User หรือ ผู้ดูแลระบบนั้นคือ User ที่มีอำนาจสูงสุดสามารถจัดการทุกส่วนในระบบรวมถึง ระบบรักษาความปลอดภัยต่างๆในระบบนั้น ในการทำหน้าที่เป็น Super User นั้นเราจะต้องทำ

การ Login โดยใช้ User Name ว่า Root หรืออาจ Login เป็นผู้ใช้ธรรมดาก็ได้แล้วใช้คำสั่ง su เพื่อเปลี่ยนตนเองเป็น Super User อีกทีหนึ่ง

2.การ Shutdown และ Reboot ระบบ

ในระบบ Unix นั้นเราสามารถที่จะปิดเครื่องหรือ Reboot เครื่องในขณะที่อยู่ที่ Shell prompt หรือที่ Login Prompt ได้ ผู้ดูแลระบบนั้นที่จะทำการ ปิดเครื่อง โดยจะต้องใช้คำสั่ง init ซึ่งอยู่ในรูปแบบดังนี้

```
init< run level >
```

จากรูปแบบข้างต้น run level จะเป็นตัวเลขที่ใช้ในการระบุว่าเราต้องการทำอะไร ถ้าเราใช้เลข 0 จะเป็นการ shutdown ระบบ ส่วนเลข 6 จะเป็นการ Reboot ตัวเลขอื่นก็จะมีความหมายอย่างอื่น แต่จะไม่กล่าวถึงในที่นี้ ปกติแล้ว Unix จะมีคำสั่งอื่นที่ทำหน้าที่เหมือน init แต่รูปแบบการใช้งานมักจะแตกต่างกันระหว่าง Unix ชนิดต่างๆ เช่นในการ Linux เราสามารถใช้คำสั่ง init 0 และ init 6 ดังตัวอย่าง

```
shutdown –h  now
reboot
```

คำสั่งแรกเป็นการ Shutdown ระบบ คำว่า now หมายถึง Shutdown ทันทีไม่ต้องรอส่วนคำสั่ง Reboot นั้นมีความหมายตรงตัวคือจะทำให้เครื่องทำการ Reboot ใหม่

3. การจัดการ User และ Group

ในส่วนนี้ เราจะศึกษาถึงการเพิ่มและลบ User และ Group ของระบบ

3.1 User Add ใช้ในการเพิ่มผู้ใช้เข้าไปในระบบ

เมื่อเราต้องการให้ใครสามารถเข้ามาใช้ระบบของเราได้ เราจะต้องเพิ่มชื่อของเขาเข้าไปในรายการ User ของระบบ โดยเราจะใช้คำสั่ง User Add ซึ่งมีรูปแบบทั่วๆไปดังนี้

```
useradd –u <uid> -g <group> -d <home dir> -s <shell> -m login-name
```

โดยการ Option แต่ละส่วนมีความหมายดังนี้

-u<uid> กำหนดหมายเลข User ID ซึ่งเป็นหมายเลขประจำตัวผู้ใช้ทุกคน

	และจะไม่ซ้ำกัน
-g<group>	กำหนด Group ให้กับ User
-d<home dir>	กำหนด Home Directory
-s<shell>	กำหนด Shell ให้กับ User
-m	บอกให้ทำการสร้าง Home Directory ขึ้นมาถ้ายังไม่มี
login-name	คือชื่อ Login ของผู้ใช้ใหม่

ใน Unix บางตัวอาจใช้ Option ที่แตกต่างจากนี้ไปก็ได้ ต่อไปนี้เป็นตัวอย่างการใช้คำสั่ง

useradd

useradd –u 120 –g teacher –d /home/arnan –s /bin/sh –m arnan คำสั่งเบื้องต้นเป็นการเพิ่ม user ที่มีชื่อ arnan เข้าไปในระบบโดยมี user-id=120 ,อยู่ในกลุ่ม teacher ,Home directory อยู่ที่/home/arnan และใช้ Broune Shell เมื่อเราเพิ่ม User เข้าไปในระบบโดยใช้คำสั่ง useradd แล้ว User นั้นจะยังไม่สามารถเข้ามาใช้งานได้เพราะเรายังไม่กำหนด password ให้กับผู้ใช้ใหม่ การกำหนด password นั้นเราจะใช้คำสั่ง passwd เช่น

```
alphar : ~#  passwd arnan
Changing password for arnan
Enter new password :
Re-type new password :
```

การสร้าง User นี้จริงๆแล้วเป็นการเพิ่มรายการใหม่เข้าไปในไฟล์ที่ชื่อ /etc/passwd ไฟล์ๆ นี้มีลักษณะดังแสดงในรูปต่อไปนี้

```
root : SswKjrA3vsNlo : 0 : 0 : root :/root :/bin/bash
bin : * : 1 : 1 : bin : /bin :
daemon : * : 2 : 2 : daemon : /sbin :
adm: * : 3 : 4 : adm :/var/adm :
lp : * : 4 : 7 : lp : /var/spool/lpd :
sync : * : 5 : 0 : sync : /sbin : /bin/sync
shutdown : * : 6 : 0 :shutdown : /sbin : /sbin/shutdown
postmaster : * :14 : 12 : postmaster : /var/spool/mail : /bin/bash
nobody : * : -1 : 100 : nobody : /dev/null :
ftp : * : 404 :1 : : /home/ftp:/bin/bash
guest : z9nQ6gr5DAD3c : 405 : 100 :guest :/home/guest :/bin/bash
```

arnan : ZJyeBGNF56SGU : Arnan Sipitakiat : /home/arnan :/bin/bash

จากตัวอย่างข้างต้นหนึ่งบรรทัดก็จะแทนข้อมูลของผู้ใช้หนึ่งคน ซึ่งในแต่ละบรรทัดก็จะแบ่งออกเป็นหลายๆส่วนถ้าเราไม่ต้องการใช้คำสั่ง useradd เราสามารถเข้าไปแก้ไฟล์ /etc/passwd นี้โดยตรงแทนก็ได้

3.2 userdel ใช้ลบ User

ถ้าเราต้องการลบ User คนใดทิ้งเราสามารถทำได้โดยการใช้คำสั่ง userdel เช่น userdel arnan จากตัวอย่างข้างต้นเป็นการลบ User ที่ชื่อ arnan ออกไปจากระบบ ถ้าเราไม่ใช้คำสั่ง userdel เราสามารถลบ User ได้โดยการลบบรรทัดในไฟล์ /etc/passwd ก็ได้

3.3 การเปลี่ยนแปลงรายละเอียดของ User และการ Disable user

ถ้าเราต้องการเปลี่ยนแปลงรายละเอียดบางอย่างของ User เช่น Home Directory หรือ Shell ที่ใช้ เราสามารถทำได้โดยการเข้าไปแก้ในไฟล์/etc/passwd แต่ใน Unix บางระบบจะมีคำสั่งที่ทำหน้าที่นี้โดยเฉพาะให้ได้แก่คำสั่ง usemod เป็นต้น

ส่วนการ Disable User นั้นเราจะทำเมื่อเราต้องการยับยั้งการเข้าใช้งานของ User บางคนไว้ชั่วคราวโดยยังไม่ลบ User นั้นทิ้ง วิธีง่ายที่สุดที่เราสามารถทำได้คือ เปลี่ยน password ของ User นั้นใหม่ แต่ก็จะทำให้แก้ password เดิมของ User นั้นหายไป อีกวิธีหนึ่งที่ใช้คือการเข้าไปแก้ในไฟล์ /etc/password โดยอาจเพิ่มตัวอักษรอะไรสักตัวหนึ่งเข้าไปใน password เดิม เช่น จากรูปใน 4.3.1 password ของ User ที่ชื่อ arnan ถูกเก็บไว้เป็น ZJyeBGNF56SGU password นี้เป็น password ที่ถูกเข้ารหัสไว้แล้ว เราสามารถเพิ่มตัวอักษรเข้าไปได้เช่น !ZJyeBGNF56SGU ตัวอย่างข้างต้นเราใส่เครื่องหมาย ! นำหน้า password เดิม ซึ่งจะทำให้ password ทั้งหมดเปลี่ยนแปลงไป เมื่อเราต้องการให้ password เก่ากลับมาเราก็ทำได้โดยการลบเครื่องหมาย ! นี้ทิ้ง

4. การ Backup ข้อมูล

โดยปกติในระบบ Unix ที่มีผู้ใช้มากๆผู้ดูแลระบบจะต้องทำการ Backup ข้อมูลของระบบไว้โดยการ Backup นั้นสามารถทำได้โดยใช้คำสั่ง แต่คำสั่งที่เป็นที่นิยมคือคำสั่ง tar

4.1 tar ใช้ในการ Backup ข้อมูล

คำสั่งนี้จะทำการรวบรวมไฟล์หลายๆไฟล์เข้าไว้ด้วยกันเป็นไฟล์เดียว ซึ่งมีรูปแบบการใช้งานดังนี้

```
tar –cvf<tar filename><files to tar>
```

คำสั่งข้างต้นจะเป็นการสร้างไฟล์ tar ขึ้นมา เช่น tar -cvd test..tar *

ตัวอย่างข้างต้นนี้จะทำการ Copy ไฟล์ทุกไฟล์ในไดเร็กทอรีปัจจุบันรวมทั้งไฟล์ในไดเร็กทอรีย่อย มาไว้ในไฟล์ที่ชื่อว่า test.tar ถ้าเราต้องการขยายไฟล์ออก เราก็สามารถทำได้โดยใช้คำสั่ง tar ในรูปแบบต่อไปนี้

```
tar –xvf<tar file>
```

ยกตัวอย่างเช่น tar test . tar

จะเป็นการขยายไฟล์ต่างๆใน test .tar ออกมา

4.2 การย่อขนาดไฟล์

ในการใช้คำสั่ง tar นั้นจะเป็นการรวมไฟล์เข้าด้วยกันเท่านั้น แต่ขนาดของมันไม่ได้ถูกย่อลงเลย แต่ในการ Backup ข้อมูลนั้นเรานิยมที่จะทำการย่อข้อมูลก่อนเก็บเพื่อประหยัดเนื้อที่ ใน UNIX จะมีโปรแกรมที่ใช้ย่อขนาดไฟล์ที่เป็นที่นิยมอยู่สองตัวคือ Compress และ gzip ซึ่งมีรูปแบบการใช้งานดังนี้

```
compress<file to compress>
gzip<file to compress>
```

ไฟล์ที่ได้จากการใช้คำสั่ง compress จะมีนามสกุล .Z ตามท้าย ส่วน gzip จะได้ไฟล์ที่มีนามสกุล .gz ตามท้าย โดยปกติแล้วไฟล์ที่ย่อโดยใช้ gzip จะมีขนาดเล็กกว่าไฟล์ที่ใช้ Compress

ในกรณีที่เราต้องการขยายไฟล์กลับคืน เราจะใช้คำสั่งในรูปแบบต่อไปนี้

```
uncompress<.Z file>
gzip-d<.gz file> or gunzip<.gz file>
```

5. การจัดการ File System และ Disk

คำสั่งในหัวข้อนี้จะเกี่ยวกับ Disk ในระบบ ซึ่งมักจะหมายถึงการเพิ่ม Disk ถ้าเราต้องการเพิ่มพื้นที่ Disk ในระบบของเรา โดยการใส่ Disk ตัวใหม่เข้าไปในสิ่งต่างๆที่จะต้องทำคือ

1. ทำการแบ่ง partition ตามต้องการ
2. สร้าง File System ขึ้นมาบน partition
3. ทำการติดตั้ง File System นั้นเข้าสู่ระบบโดยการ mount disk

ในขั้นตอนของการแบ่ง partition นั้นอาจข้ามไปก็ได้ถ้าไม่ต้องการแบ่งมัน คำสั่งที่ใช้ในการแบ่ง partition ก็ได้แก่คำสั่ง fdisk

5.1 การสร้าง File System

เมื่อเราแบ่ง partition เรียบร้อยแล้ว ต่อไปก็ต้องสร้าง file system ขึ้นมาบนแต่ละ partition โดย File System นั้นมีอยู่หลายแบบแล้วแต่ว่าเราจะเลือกใช้แบบไหน เช่น UFS, DOS-FAT,EXT2 เป็นต้น UNIX แต่ละตัวก็จะรู้และใช้งาน File System ได้หลายชนิด ซึ่งขึ้นอยู่กับผู้ดูแลระบบว่าจะเลือกใช้แบบใด คำสั่งที่ใช้ในการสร้าง File System นั้น มักจะต่างกันสำหรับ UNIX แต่ละตัวแต่โดยทั่วไปมักจะใช้คำสั่ง mkfs (make file system) ซึ่งมีรูปแบบการใช้งานดังนี้

mkfs –t<file system type><device>

โดยที่ File System Type คือชนิดของ File System ที่ต้องการจะสร้าง ซึ่งปกติแล้วสามารถหาดูได้จาก /etc/fs ส่วน Device นั้นหมายถึงชื่อ Device ของ Disk ที่ต้องการ ชื่อนี้จะอยู่ภายใต้ directory /dev และ Unix แต่ละตัวก็จะใช้ชื่อที่แตกต่างกัน เช่น ใน Linux จะใช้ /dev/hda สำหรับ Harddisk ตัวที่ 1 และ /dev/hdb สำหรับตัวที่สอง และ /dev/hdc สำหรับตัวที่สาม ไปเรื่อยๆใน Harddisk แต่ละตัวก็สามารถมีได้หลาย partition จึงแบ่งเป็น /dev/hdax โดย x แทนเลขของ partition เช่น /dev/hda1 และ /dev/hda2 แทน Harddisk ตัวที่หนึ่ง partition ที่ 1 และ 2 ตามลำดับเป็นต้น

mkfs –t ext2 /dav/hda2

ตัวอย่างคำสั่งข้างต้นเป็นการสร้าง File System ชนิดที่ชื่อ EXT2 ขึ้นมาโดยใช้ Harddisk ตัวที่ 1 partition ที่ 2

5.2 การ Mount และ Unmount File System

เมื่อเราทำการสร้าง File System ขึ้นมาแล้วสิ่งสุดท้ายที่จะต้องทำเพื่อให้ระบบสามารถใช้งาน Disk นั้น ได้คือ การ mount มันเข้ามาในระบบ ใน UNIX นั้นจะไม่มอง Disk แต่ละตัวเป็น Drive แยกต่างหากเหมือนใน DOS แต่จะมองเป็นไดเร็กทอรีแทน โดยเราจะใช้คำสั่ง mount ในการกำหนดว่า ต้องการให้ Disk ไปปรากฏอยู่ที่ไดเร็กทอรีใด โดยมีรูปแบบการใช้งานดังนี้

```
mount –t<file system type><drvice name><mount point>
```

โดย<file system type>ก็คือชนิดของ File System ที่ต้องการ mount ส่วน device name คือชื่อของ Disk นั้นๆ และ mount point หมายถึงไดเร็กทอรี ที่ต้องการให้ disk นั้นไปปรากฏ โดยไดเร็กทอรีนั้นจะต้องมีอยู่แล้วและว่างอยู่ไม่มี Disk อื่น mount อยู่ก่อนหน้า

mount –t msdos /dev/hda1 /mnt

ตัวอย่างข้างต้นเป็นการ mount Harddisk แรก partition แรก ซึ่งเป็น partition ของ DOS (ในกรณีที่เครื่องนั้นมีทั้ง DOS และ UNIX) มาไว้ที่ directory /mnt เมื่อเราทำคำสั่งนี้เสร็จแล้ว Drive C ของ DOS ก็จะมาปรากฏอยู่ใน directory /mnt ของ UNIX ถ้าเราต้องการดูว่าในปัจจุบันเรา mount อะไรไว้บ้างก็ทำได้โดยการใช้คำสั่ง mount

เช่น

```
alphar : ~# mount
/dev/hda3 on / type ext2 (rw)
/dev/hda1 on / dosc type msdos (rw)
none on / proc type proc(rw)
```

ในทางตรงกันข้ามถ้าเราต้องการที่จะเอา Disk ตัวใดออกจากระบบ เราจะใช้คำสั่ง unmount ซึ่งมีรูปแบบการใช้งานดังนี้

```
unmount<device name or mount point>
```

พารามิเตอร์ที่ต้องใส่ไปคือ ชื่อ device ของ Disk หรือไดเร็กทอรีที่ทำการ mount disk นั้นเข้ามา ยกตัวอย่างเช่นunmount /mnt

ตัวอย่างข้างต้นเป็นการ unmount disk mount อยู่ที่ directory /mnt