การสร้างเว็บเซิร์ฟเวอร์ด้วยลีนุกซ์คลัสเตอร์

Web Server Using Linux Clustering

นายสิรเชษฐ์ ลี้ไพโรจน์กุล
นายอนันศักดิ์ จตุพรสมบัติ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต
ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2543

การสร้างเว็บเซิร์ฟเวอร์ด้วยลีนุกซ์คลัสเตอร์

Web Server Using Linux Clustering

โดย

นายสิรเชษฐ์ ลี้ไพโรจน์กุล

นายอนันศักดิ์ จตุพรสมบัติ

อาจารย์ที่ปรึกษา

ผศ. บรรจง ปิยธำรง

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2543

ปริญญานิพนธ์ปีการศึกษา 2543

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การสร้างเว็บเซิร์ฟเวอร์ด้วยลีนุกซ์คลัสเตอร์

Web Server Using Linux Clustering

ผู้จัดทำ

1. นายสิรเชษฐ์ ลี้ไพโรจน์กุล รหัสประจำตัว 40010853
2. นายอนันศักดิ์ จตุพรสมบัติ รหัสประจำตัว 40010937

_____ อาจารย์ที่ปรึกษา

(ผศ. บรรจง ปียธำรง)

การสร้างเว็บเซิร์ฟเวอร์ด้วยลีนุกซ์คลัสเตอร์

นายสิรเชษฐ์ ลี้ไพโรจน์กุล 40010853
นายอนันศักดิ์ จตุพรสมบัติ 40010937
ผศ. บรรจง ปิยธำรง อาจารย์ที่ปรึกษา
ปีการศึกษา 2543

บทคัดย่อ

ระบบคลัสเตอร์เป็นแนวทางหนึ่งในการแก้ปัญหาหลายๆปัญหา เช่น งานที่ต้องการการคำนวณที่ค่อนข้างสูง ตัวอย่างของระบบนี้คือ ระบบคลัสเตอร์แบบแบวูล์ฟ และในงานอีกอย่างหนึ่งที่ต้องการความมีเสถียรภาพในการทำงาน ตัวอย่างของงานนี้เช่น เว็บเซิร์ฟเวอร์ ในปัจจุบันบางเว็บไซต์ที่มีผู้นิยมเยี่ยมชมมาก อาจต้องประสบกับปัญหาเว็บเซิร์ฟเวอร์ไม่สามารถรองรับสัญญาณร้องขอที่มีเข้ามามากเกินไปได้ หนทางหนึ่งในการแก้ปัญหานี้ก็คือ การใช้เว็บเซิร์ฟเวอร์หลายๆเครื่องช่วยกันบริการ ซึ่งใช้หลักการของการกระจายโหลดไปยังหลายๆเครื่องนั่นเอง และวิธีการที่ใช้ในการกระจายโหลดก็มีอยู่ด้วยกันหลายวิธีด้วยกัน เช่น DNS Round-Robin , Reverse Proxy Server , Linux Virtual Server ซึ่งวิธีการสุดท้ายเป็นวิธีที่เลือกใช้ในการทำโครงงานครั้งนี้

Web Server Using Linux Clustering

Sirachet Lipairotkun

Anansak Jatupornsombut

Assist. Prof. Bunjong Piyathumrong

Advisor

ABSTRACT

Cluster system is the solution for very situation. Such as the job that need very high computing, beowulf is the example of this situation. Or job that need high availability , high reliability and can be cover fail over ,web server is needed by this job. And this is the objective of this project. Nowadays in some web site that very popular ,may be receive many million requests per day. It will make web server to be overloading. One way to resolve this problem is using the concept of load balancing for many servers. There are many solutions to balance load , such as DNS round-robin , reverse proxy server , linux virtual server. The last solution is concentrate in this project.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้คงไม่อาจเสร็จได้ด้วยดี หากไม่ได้รับความช่วยเหลือ และร่วมมือจากหลายๆ ฝ่ายด้วยกัน บุคคลแรกที่ต้องกล่าวถึงเพราะเป็นส่วนสำคัญที่ทำให้วิทยานิพนธ์นี้เสร็จลงได้ก็คือ ผศ.บรรจง ปิยธำรง อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ให้ความเอาใจใส่ แนะนำ และช่วยเหลือเสมอมา ซึ่งต้องขอขอบพระคุณเป็นอย่างมาก

และต้องขอขอบพระคุณบุคคลสำคัญที่สุดที่ทำให้ข้าพเจ้ามีวันนี้ ก็คือ บิดา มารดา อันเป็นที่เคารพรักยิ่ง ซึ่งได้เลี้ยงดูผู้เขียนมาเป็นอย่างดี พร้อมทั้งให้โอกาสในการศึกษาอย่างเต็มที่ และยังให้กำลังใจเอาใจใส่เสมอมา ในทุกๆด้านอันหาที่เปรียบมิได้ ข้าพเจ้าขอระลึกในพระคุณอันสุดประมาณ และขอกราบขอบพระคุณมา ณ ที่นี้

สิรเชษฐ์ ลี้ไพโรจน์กุล
อนันศักดิ์ จตุพรสมบัติ

สารบัญ

			หน้าที่
บทคัดย่อภาษาไทย			I
บทคัดย่อภาษาอังกฤษ			II
กิตติกรรมประกาศ			III
สารบัญ			IV
สารบัญตาราง			VI
สารบัญรูปภาพ			VII
บทที่	1	บทนำ	
	1.1	ความสำคัญและที่มา	1
	1.2	วัตถุประสงค์ของโครงงาน	2
	1.3	ขอบเขตของงานวิจัย	3
	1.4	ประโยชน์ที่คาดว่าจะได้รับ	3
	1.5	วิธีการดำเนินงาน	3
		1.5.1 ขั้นตอนการศึกษาหาข้อมูล	3
		1.5.2 ขั้นตอนการติดตั้งระบบ	3
		1.5.3 ขั้นตอนทดลองและปรับแต่ง	3
		1.5.4 ขั้นตอนการสรุปผล	4
บทที่	2	Web Server Clustering	
	2.1	เว็บเซิร์ฟเวอร์	5
	2.2	Web Server Clustering แบบต่างๆ	5
	2.3	ใช้ DNS Round-Robin	6
		2.3.1 หลักการของ DNS Round-Robin	6
		2.3.2 การติดตั้ง Domain Name Server เพื่อแบ่งโหลดให้เว็บเซิร์ฟเวอร์	8
	2.4	ใช้ Reverse Proxy	9
		2.4.1 หลักการของ Reverse Proxy	9
		2.4.2 การติดตั้ง Reverse Proxy Server เพื่อแบ่งโหลดให้เว็บเซิร์ฟเวอร์	11
บทที่	3	Linux Virtual Server	
	3.1	สถาปัตยกรรมโดยรวมของระบบ Linux Virtual Server	12
	3.2	เทคนิคทางด้าน IP Load Balancing	15
		3.2.1 Linux Virtual Server via NAT (Network Address Translation)	15
		3.2.2 Linux Virtual Server via IP Tunneling	19

		3.2.3 Linux Virtual Server via Direct Routing	23
		3.2.4 ข้อเปรียบเทียบของวิธีการต่างๆ	25
		3.2.5 วิธีการทำงานภายในระบบ	26
	3.3	Connection Scheduling	27
		3.3.1 Round – Robin Scheduling	28
		3.3.2 Weighted Round – Robin Scheduling	28
		3.3.3 Least - Connection Scheduling	29
		3.3.4 Weighted Least – Connection Scheduling	30
บทที่	4	การติดตั้ง	
	4.1	Apache Web Server	31
	4.2	การติดตั้งระบบ Linux Virtual Server โดยการ patch kernel	36
	4.3	การติดตั้งระบบ Linux Virtual Server โดยใช้โปรแกรม Piranha Web Interface	39
บทที่	5	ผลการทดสอบ	53
บทที่	6	บทวิจารณ์และสรุปผล	
	6.1	สรุปผลการทดสอบ	60
	6.2	บทวิจารณ์	64
บรรณานุกรม			65

สารบัญตาราง

หน้าที่

ตารางที่ 3.1 แสดงตัวอย่างกฎที่กำหนดการทำงานของ Linux Virtual Server via NAT 18
ตารางที่ 3.2 แสดงตัวอย่างกฎที่เป็นการติดตั้งแบบ Linux Virtual Server via IP Tunneling 22
ตารางที่ 3.3 แสดงข้อเปรียบเทียบของวิธีการต่างๆที่ใช้ใน Linux Virtual Server 25
ตารางที่ 6.1 แสดงค่าเฉลี่ยของจำนวน Req./sec. และอัตรา Throughput เมื่อเพิ่ม Real server 60
ตารางที่ 6.2 แสดงค่าเฉลี่ยของจำนวน Req./sec. และอัตรา Throughput ของแต่ละอัลกอริทึม 62

สารบัญรูปภาพ

หน้าที่

รูปที่ 2.1 แสดงการใช้ Domain Name Server แบ่งโหลดให้เว็บเซิร์ฟเวอร์ 7
รูปที่ 2.2 แสดงการใช้ Reverse Proxy Server แบ่งโหลดให้เว็บเซิร์ฟเวอร์ 10
รูปที่ 3.1 แสดงสถาปัตยกรรมโดยรวมของระบบ Linux Virtual Server 12
รูปที่ 3.2 แสดงสถาปัตยกรรมของระบบ Linux Virtual Server via NAT 16
รูปที่ 3.3 แสดงตัวอย่างการติดตั้งของระบบ Linux Virtual Server via NAT 17
รูปที่ 3.4 แสดงสถาปัตยกรรมของระบบ Linux Virtual Server via IP Tunneling 20
รูปที่ 3.5 แสดงขั้นตอนการทำงานของ Linux Virtual Server via IP Tunneling 21
รูปที่ 3.6 แสดงสถาปัตยกรรมของระบบ Linux Virtual Server via Direct Routing 23
รูปที่ 3.7 แสดงขั้นตอนการทำงานของ Linux Virtual Server via Direct Routing 24
รูปที่ 3.8 แสดงการทำงานภายในของระบบ Linux Virtual Server 27
รูปที่ 4.1 แสดงหน้าจอของ CONTROL/MONITORING Panel 42
รูปที่ 4.2 แสดงหน้าจอของ GLOBAL SETTINGS Panel 43
รูปที่ 4.3 แสดงหน้าจอเริ่มแรกของ Redundancy Panel 44
รูปที่ 4.4 แสดงหน้าจอ Redundancy Panel ที่ใช้ในการติดตั้ง 45
รูปที่ 4.5 แสดงหน้าจอเริ่มแรกของ Failover Panel 46
รูปที่ 4.6 แสดงหน้าจอของ Failover Panel ที่ได้ทำการติดตั้ง 47
รูปที่ 4.7 แสดงหน้าจอ Failover Panel ที่ได้ทำการแก้ไข 48
รูปที่ 4.8 แสดงหน้าจอ Failover Panel ที่ใช้แก้ไข script 49
รูปที่ 4.9 แสดงหน้าจอเริ่มแรกของ VIRTUAL SERVERS Panel 50
รูปที่ 4.10 แสดงหน้าจอ VIRTUAL SERVERS Panel ที่ได้ทำการเพิ่มเครื่องเซิร์ฟเวอร์ 51
รูปที่ 4.11 แสดงหน้าจอ VIRTUAL SERVERS Panel ที่แก้ไขค่าติดตั้งของเครื่องเซิร์ฟเวอร์ 52
รูปที่ 5.1 แสดงค่า Req./sec ของเว็บเซิร์ฟเวอร์ 1 เครื่อง 54
รูปที่ 5.2 แสดงค่า Throughput ของเว็บเซิร์ฟเวอร์ 1 เครื่อง 54
รูปที่ 5.3 แสดงค่า Req./sec ของ LVS – wlc 1 เครื่อง 55
รูปที่ 5.4 แสดงค่า Throughput ของ LVS – wlc 1 เครื่อง 55
รูปที่ 5.5 แสดงค่า Req./sec. ของ LVS – wlc 2 เครื่อง 56
รูปที่ 5.6 แสดงค่า Throughput ของ LVS – wlc 2 เครื่อง 56
รูปที่ 5.7 แสดงค่า Req./sec. ของ LVS – wlc 3 เครื่อง 57

รูปที่ 5.8 แสดงค่า Throughput ของ LVS – wlc 3 เครื่อง 57
รูปที่ 5.9 แสดงค่า Req./sec. ของ LVS – rr 3 เครื่อง 58
รูปที่ 5.10 แสดงค่า Throughput ของ LVS – rr 3 เครื่อง 58
รูปที่ 5.11 แสดงค่า Req./sec. ของ LVS – wrr 3 เครื่อง 59
รูปที่ 5.12 แสดงค่า Throughput ของ LVS – wrr 3 เครื่อง 59
รูปที่ 6.1 แสดงการเปรียบเทียบจำนวน Req./sec. เมื่อเพิ่มจำนวน Real server 61
รูปที่ 6.2 แสดงการเปรียบเทียบอัตรา Throughput เมื่อเพิ่มจำนวน Real server 61
รูปที่ 6.3 แสดงการเปรียบเทียบจำนวน Req./sec. ของแต่ละอัลกอริทึม 63
รูปที่ 6.4 แสดงการเปรียบเทียบอัตรา Throughput ของแต่ละอัลกอริทึม 63

บทที่ 1
บทนำ

1.1 ความสำคัญและที่มา

ในปัจจุบันการบริการทางอินเตอร์เน็ตมีการใช้งานอย่างกว้างขวางและเติบโตเพิ่มมากขึ้นอย่างรวดเร็วโดยเฉพาะการบริการทางด้านเอกสารอิเล็กทรอนิกส์ ซึ่งให้บริการโดยโปรแกรมเว็บเซิร์ฟเวอร์ และเมื่อมีการใช้งานทางอินเตอร์เน็ตที่มากขึ้นประกอบกับจำนวนของผู้ใช้บริการทางอินเตอร์เน็ตมีอัตราเพิ่มมากขึ้นอย่างรวดเร็ว ซึ่งก็จะทำให้เครื่องเซิร์ฟเวอร์ที่ให้บริการทางอินเตอร์เน็ตมีการทำงานที่เพิ่มมากขึ้น ส่งผลให้การบริการเพื่อตอบสนองความต้องการที่มีมากขึ้นนั้นเป็นไปอย่างล่าช้า และเนื่องด้วยมาจากจำนวนผู้ใช้งานทางอินเตอร์เน็ตและภาระงานที่เครื่องเซิร์ฟเวอร์ต้องทำมีจำนวนเพิ่มมากขึ้น เป็นเหตุให้เกิดแนวความคิดว่าจะทำอย่างไรให้ระบบเซิร์ฟเวอร์ที่ใช้งานอยู่สามารถเติบโตไปพร้อมกับความต้องการที่มีเพิ่มมากขึ้นอย่างต่อเนื่องนี้ได้ อีกทั้งระบบเซิร์ฟเวอร์จะต้องทำงานบริการทางอินเตอร์เน็ตได้อย่างต่อเนื่องและมีความน่าเชื่อถือในการใช้งาน โดยที่วิธีการนั้นจะต้องอาศัยทั้งทางซอฟแวร์และฮาร์ดแวร์เข้ามาช่วยเพื่อให้ระบบทำงานเป็นไปตามความต้องการนี้ ซึ่งปัจจัยที่ต้องคำนึงถึงในการสร้างระบบนี้คือ

- *Scalability* เป็นความต้องการที่เมื่อระบบที่เราใช้อยู่มีภาระงานทางด้านการบริการทางอินเตอร์เน็ตที่เพิ่มมากขึ้น เราสามารถที่จะทำการปรับแต่งระบบของเราให้สามารถรองรับกับความต้องการที่เปลี่ยนไปนี้ได้อย่างง่าย ไม่มีความซับซ้อน
- *24x7 Availability* ระบบจะต้องสามารถทำงานได้อย่างต่อเนื่องแม้ว่าจะมีการทำงานที่ผิดพลาดอันเนื่องมาจากปัญหาทางซอฟแวร์หรือฮาร์ดแวร์ก็ตาม
- *Manageability* ระบบที่ใช้สามารถทำการจัดการดูแลระบบได้ง่าย แม้ว่าระบบที่ใช้จะมีขนาดใหญ่ก็ตาม
- *Cost-effectiveness* ระบบทั้งหมดมีค่าใช้จ่ายในการสร้างที่ไม่แพง และสามารถที่จะขยายระบบได้ง่าย

การทำงานของเซิร์ฟเวอร์ในแบบที่เป็นเครื่องเดียวนั้นจะไม่สามารถรองรับกับความต้องการข้างต้นได้ การปรับปรุงเปลี่ยนแปลงระบบเป็นไปอย่างยากลำบาก เพราะว่าจะต้องทำการเปลี่ยนเครื่องเซิร์ฟเวอร์ให้เป็นเครื่องที่มีประสิทธิภาพสูงกว่าเดิม และเมื่อใช้งานไปได้สักระยะหนึ่งแล้วเกิดความต้องการในการใช้งานมีเพิ่มมากขึ้นอีก ซึ่งเครื่องเซิร์ฟเวอร์เครื่องนั้นไม่สามารถรองรับได้ก็จะต้องทำการเปลี่ยนเครื่องเซิร์ฟเวอร์ให้เป็นเป็นเครื่องที่มีประสิทธิภาพมากกว่าเดิมอีก ส่งผลให้เสียค่าใช้จ่ายเป็นจำนวนมากในการปรับปรุงเปลี่ยนแปลงระบบในแต่ละครั้งที่ทำการเปลี่ยนแปลง อีกประเด็นหนึ่งก็คือเครื่องเซิร์ฟเวอร์แบบเครื่องเดียวไม่สามารถรองรับกับกรณีที่เครื่องเซิร์ฟเวอร์นั้นเกิดหยุดทำงานขึ้นมาอันเนื่องมาจากความผิดพลาดทางซอฟแวร์หรือฮาร์ดแวร์อย่างใดอย่างหนึ่งหรือทางสองอย่าง ส่งผลให้การบริการทางอินเตอร์เน็ตของเซิร์ฟเวอร์เครื่องนั้นต้องหยุดชะงักลงไม่สามารถให้บริการได้อีก ทำให้เว็บ

ไซด์นั้นไม่เป็นที่น่าเชื่อถือต่อผู้ใช้บริการ และยังเกิดผลเสียหายอย่างมากต่อธุรกิจที่ทำผ่านทางอินเตอร์เน็ตอีกด้วย

เทคโนโลยีทางคลัสเตอร์ได้มีการพัฒนาไปมาก มีการนำเทคโนโลยีทางคลัสเตอร์มาใช้ในงานต่างๆอย่างหลากหลาย ไม่ว่าจะเป็นการทำเซิร์ฟเวอร์แบบที่มีการสำรอง โดยเครื่องที่ทำหน้าที่สำรองจะทำงานแทนเครื่องเซิร์ฟเวอร์หลักเมื่อเครื่องเซิร์ฟเวอร์หลักไม่สามารถทำงานได้อันเนื่องมาจากความผิดพลาดในการทำงานของเครื่องเซิร์ฟเวอร์หลัก หรือการนำมาใช้ในงานคำนวณทางด้านวิทยาศาสตร์ที่ต้องอาศัยการทำงานแบบขนาน เช่น การคลัสเตอร์แบบ Beowulf ซึ่งเป็นวิธีการที่ใช้คอมพิวเตอร์ส่วนบุคคลจำนวนมากมาต่อเป็นระบบคลัสเตอร์เพื่อการคำนวณที่ต้องใช้เวลาในการคำนวณมาก เช่น งานทางด้านการพยากรณ์อากาศ เป็นต้น ซึ่งวิธีการคลัสเตอร์ที่นำมาใช้ในการบริการทางอินเตอร์เน็ตนั้นจะคำนึงถึงเรื่องของระบบที่สามารถทำการปรับเปลี่ยนให้สามารถรองรับกับการความต้องการบริการที่เพิ่มมากขึ้นได้ ประกอบกับระบบมีความคงทนต่อการผิดพลาดที่เกิดขึ้นภายในระบบ และยังคงทำงานต่อไปได้แม้จะมีความผิดพลาดเกิดขึ้น ซึ่งสถาปัตยกรรมแบบคลัสเตอร์นี้จะเป็นแบบ loosely couple ซึ่งก็คือสามารถทำการปรับปรุงเปลี่ยนแปลงระบบได้ง่าย และประหยัดค่าใช้จ่ายในการเปลี่ยนแปลงระบบ รวมทั้งยังมีความน่าเชื่อถือกว่าระบบที่เป็นแบบหลายโพรเซสเซอร์ที่เป็นแบบ tightly couple ซึ่งยากต่อการเปลี่ยนแปลงระบบ

ด้วยเหตุนี้จึงเป็นที่มาของการทำโครงงานนี้ขึ้น เพื่อศึกษาการนำเทคโนโลยีทางด้านการคลัสเตอร์มาใช้กับเว็บเซิร์ฟเวอร์เพื่อให้รองรับกับความต้องการในการใช้งานทางอินเตอร์เน็ตที่มีมากขึ้นให้เป็นไปอย่างมีประสิทธิภาพและสามารถปรับปรุงระบบให้มีประสิทธิภาพมากขึ้นเป็นไปโดยไม่ยุ่งยาก รวมทั้งศึกษาถึงวิธีการที่จะทำให้ระบบสามารถทำงานได้อย่างต่อเนื่องแม้ว่าจะมีความผิดพลาดเกิดขึ้นก็ตาม (Fault Tolerance) คือระบบสามารถทำงานต่อไปได้แม้ว่าระบบจะมีความผิดพลาดเกิดขึ้น และทางกลุ่มได้เลือกระบบปฏิบัติการลีนุกซ์ในการทำงานเป็นเซิร์ฟเวอร์เพราะว่าเป็นระบบปฏิบัติการที่ใช้งานได้ฟรีและเป็นระบบปฏิบัติการที่มีประสิทธิภาพ และเป็นที่นิยมใช้ในการทำเป็นเซิร์ฟเวอร์ที่ให้บริการทางอินเตอร์เน็ต

1.2 วัตถุประสงค์ของโครงงาน

1.2.1 เพื่อศึกษาหลักการทำงานของโปรแกรมเว็บเซิร์ฟเวอร์

1.2.2 สามารถปรับแต่งการทำงานของโปรแกรมเว็บเซิร์ฟเวอร์ได้

1.2.3 เพื่อศึกษาเทคโนโลยีทางด้านการคลัสเตอร์

1.2.4 ศึกษาวิธีการในการนำเทคโนโลยีทางคลัสเตอร์มาใช้ในการทำงานร่วมกันกับเว็บเซิร์ฟเวอร์ เพื่อให้ได้ระบบที่มีความสามารถทางด้าน Scalability, 24x7 Availability, Manageability, Cost-effectiveness และ Fault Tolerance

1.2.5 สามารถเปรียบเปรียบข้อดีข้อเสียของแต่ละวิธีที่นำมาใช้ในการสร้างระบบ

1.2.6 สามารถติดตั้งระบบขึ้นมาทำงานเป็นเว็บเซิร์ฟเวอร์โดยการใช้วิธีการทางคลัสเตอร์ได้

1.2.7 สามารถทดสอบระบบที่ทำการติดตั้งได้

1.3 ขอบเขตของงานวิจัย

การทดลองและวิจัยของโครงงานครั้งนี้ขอบเขตอยู่ที่สามารถสรุปผลการทดลองได้ และทำการศึกษาหลักการทำงานที่ใช้เทคโนโลยีทางด้านการคลัสเตอร์มาใช้ในการทำงานกับเว็บเซิร์ฟเวอร์ รวมทั้งสามารถทำการติดตั้งระบบเว็บเซิร์ฟเวอร์โดยวิธีการทางคลัสเตอร์ที่ใช้ระบบปฏิบัติการเป็นลีนุกซ์ และทำการทดสอบระบบที่ได้ทำการติดตั้งได้

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1.4.1 สามารถเข้าใจหลักการติดตั้งใช้งานเซิร์ฟเวอร์ และทำการปรับปรุงเปลี่ยนแปลงระบบได้

1.4.2 สามารถเข้าใจถึงประโยชน์และหลักในการนำเทคโนโลยีทางด้านคลัสเตอร์มาใช้ร่วมกับเว็บเซิร์ฟเวอร์เพื่อให้ได้ระบบที่มีประสิทธิภาพ

1.4.3 สามารถเปรียบเทียบประสิทธิภาพในการทำงานแบบต่างๆที่ได้ทำการศึกษา

1.5 วิธีการดำเนินงาน

ขั้นตอนการดำเนินงานจะแบ่งออกเป็น 5 ขั้นตอนดังนี้คือ

1.5.1 ขั้นตอนการศึกษาหาข้อมูล

ทำการรวบรวมข้อมูลต่างๆที่จะนำมาใช้ในโครงงานนี้ ซึ่งจะเป็นเรื่องเกี่ยวกับเทคนิคทางด้านการคลัสเตอร์ รวมทั้งการติดตั้งระบบเว็บเซิร์ฟเวอร์เพื่อการบริการทางอินเตอร์เน็ตที่มีประสิทธิภาพ ซึ่งแหล่งที่ค้นคว้าก็จะมาจากหนังสือในห้องสมุด และจากอินเตอร์เน็ต

1.5.2 ขั้นตอนการติดตั้งระบบ

เป็นขั้นตอนของการติดตั้งเซิร์ฟเวอร์และโปรแกรมต่างๆที่จะใช้ภายในโครงงานนี้

- การติดตั้งระบบปฏิบัติการลีนุกซ์ในเครื่องเซิร์ฟเวอร์ที่จะใช้งาน
- การติดตั้ง Apache Web Server
- การติดตั้งโปรแกรมอื่นๆที่จำเป็นจะต้องใช้งาน
- การปรับเปลี่ยนระบบเพื่อให้ทำงานเป็นเว็บเซิร์ฟเวอร์แบบคลัสเตอร์

1.5.3 ขั้นตอนทดลองและปรับแต่ง

เป็นขั้นตอนของการทำการทดลองและปรับแต่งประสิทธิภาพการทำงานของเซิร์ฟเวอร์แต่ละเครื่องให้มีประสิทธิภาพในการทำงานที่สูงที่สุด และทำการเปรียบเทียบถึงข้อดีและข้อเสียของการติดตั้งแบบต่างๆ รวมทั้งการนำไปใช้งานที่เหมาะสมกับการติดตั้งนั้นๆ

1.5.4 ขั้นตอนการสรุปผล

เป็นขั้นตอนการสรุปผลทั้งหมดของโครงงาน โดยการเปรียบเทียบให้เห็นความแตกต่างของวิธีการต่างๆที่นำมาใช้ในการทำงาน อีกทั้งการทำงานที่เหมาะสมของวิธีการต่างๆที่ได้ทำการศึกษา รวมทั้งการปรับปรุงให้ระบบมีประสิทธิภาพมากขึ้น

บทที่ 2

Web Server Clustering

2.1 เว็บเซิร์ฟเวอร์

เว็บเซิร์ฟเวอร์เป็นโปรแกรมที่ทำงานเป็น daemon อยู่บนเครื่องที่ทำหน้าที่ให้บริการ World Wide Web ซึ่งโปรแกรมที่มีผู้นิยมใช้กันมากในปัจจุบันคือ โปรแกรม Apache Web Server เนื่องจากเป็นโปรแกรมที่ใช้งานได้ง่าย และไม่เสียค่าใช้จ่ายในการซื้อโปรแกรม เพราะเป็นโปรแกรมที่มีแจกฟรีหาได้ตามอินเตอร์เน็ต daemon ของเว็บเซิร์ฟเวอร์นี้จะคอยรอรับสัญญาณการร้องขอจากเครื่องไคลเอนต์ ซึ่งบนเครื่องไคลเอนต์ก็จะมีโปรแกรมที่ใช้ติดต่อกับโปรแกรมเว็บเซิร์ฟเวอร์ เรียกว่า โปรแกรมบราวเซอร์ เช่น Netscape Navigator , Internet Explorer , Opera เป็นต้น เมื่อโปรแกรมบราวเซอร์ได้รับการตอบรับจากเว็บเซิร์ฟเวอร์แล้ว ก็จะทำการแสดงผลบนหน้าจอของผู้ใช้ ซึ่งข้อมูลที่ส่งมาส่วนมากจะอยู่ในรูปของภาษา HTML (HyperText Markup Language) ข้อดีของภาษา HTML ก็คือ สามารถจัดเรียงเอกสารได้ง่าย และยังสามารถลิงก์เชื่อมต่อระหว่างเอกสารได้อีกด้วย

2.2 Web Server Clustering แบบต่างๆ

เนื่องจากในปัจจุบัน เว็บไซต์ที่มีผู้นิยมเยี่ยมชมมาก เช่น hotmail.com , yahoo.com , microsoft.com ซึ่งอาจมีผู้เยี่ยมชมถึงหลายแสนคนต่อวัน การที่จะมีเว็บเซิร์ฟเวอร์เพียงเครื่องเดียวดูจะไม่เพียงพอที่จะรองรับบริการมากขนาดนั้นได้ เมื่อผู้ที่เข้ามายังเว็บไซต์เรา แต่กลับต้องผิดหวังกลับไป ไม่สามารถเยี่ยมชมได้ เพราะเว็บเซิร์ฟเวอร์ให้บริการมากเกินกว่าจะทำงานต่อไปได้ ก็จะทำให้ความนิยมต่อตัวเว็บไซต์ลดลงอย่างแน่นอน ซึ่งในหลายๆเว็บไซต์ได้เล็งเห็นถึงความสำคัญข้อนี้ จึงได้มีการคิดค้นหาหนทางแก้ไขปัญหานี้ให้หมดไป และหนทางที่จะทำให้เว็บเซิร์ฟเวอร์สามารถรองรับงานได้มากขึ้นนั้นก็มีอยู่ด้วยกันสองแนวทางใหญ่ๆด้วยกัน ทางหนึ่งคือการเพิ่มประสิทธิภาพของเครื่องเว็บเซิร์ฟเวอร์ให้สูงขึ้น ซึ่งแน่นอนการซื้อเครื่องระดับเซิร์ฟเวอร์แต่ละครั้งนั้นย่อมเปลืองงบประมาณมิใช่น้อย จึงได้มีทางออกอีกทางหนึ่งนั่นก็คือ การเพิ่มจำนวนเว็บเซิร์ฟเวอร์ให้มีจำนวนมากขึ้นเพื่อมาช่วยกันรองรับงาน ซึ่งก็คือ หลักการของการบาลานซ์โหลดไปยังหลายๆเครื่องนั่นเอง และในปัจจุบันก็มีการนำมาใช้ในหลายๆแบบด้วยกัน อาทิเช่น

- แบ่งโหลดโดยใช้ Domain Name Server (DNS Round-Robin)
- ใช้หลักการของ Proxy (Reverse Proxy)
- วิธีการของ Linux Virtual Server

2.3 ใช้ DNS Round-Robin

2.3.1 หลักการของ DNS Round-Robin

Domain Name System

ก่อนอื่นมาทำความเข้าใจกันถึงหลักการของระบบ Domain Name System กันก่อน โดยปกติแล้วเครื่องคอมพิวเตอร์ทั้งหลายที่อยู่บนเครือข่ายอินเตอร์เน็ตนั้น แต่ละเครื่องจะต้องมีหมายเลขประจำเครื่องซึ่งจะต้องมีค่าไม่ซ้ำกัน เพื่อให้สามารถแบ่งแยกความแตกต่างของแต่ละเครื่องได้ หมายเลขประจำเครื่องนี้เรียกว่า IP Address การติดต่อสื่อสารกันภายในเครือข่ายอินเตอร์เน็ตจะใช้หมายเลขนี้เป็นหลัก แต่ถ้าหากเราจะเข้าเว็บไซต์ใดก็ตาม และต้องคอยจำเลข IP Address ของแต่ละเครื่องแล้วล่ะก็ อินเตอร์เน็ตก็คงจะไม่สามารถเป็นที่นิยมเป็นแน่ จะเห็นได้ว่าเวลาจะเข้าเว็บไซต์ใดก็ให้ใส่ชื่อของเว็บไซต์นั้นได้เลย เช่น ใส่ www.kmitl.ac.th ก็สามารถเข้าเว็บไซต์ของสถาบันได้ แทนที่จะต้องใส่เลข 161.246.10.21 การที่สามารถใช้ชื่อแทน IP Address ได้นี้ก็เพราะระบบ Domain Name System นั่นเอง

ระบบโดเมนเนมนี้จะมีการใช้งานสองแบบคือ การแปลงจากชื่อให้เป็น IP Address เรียกว่า Forward Mapping เช่น แปลงจาก kmitl.ac.th ให้เป็น 161.246.10.21 และการแปลงจาก IP Address ให้เป็นชื่อ เรียกว่า Reverse Mapping เช่น แปลงจาก 161.246.10.21 ให้เป็น kmitl.ac.th

การที่ระบบโดเมนเนมจะสำเร็จได้นั้นจะต้องได้รับความร่วมมือจากทุกๆมุมโลก โดยภายในพื้นที่หนึ่งๆจะต้องมีผู้รับผิดชอบที่จะต้องจัดหาเครื่อง Domain Name Server ขึ้นมา เพื่อใช้เก็บข้อมูลการแปลงกลับของเครื่องคอมพิวเตอร์ต่างๆภายในพื้นที่รับผิดชอบของตน ซึ่งจะแยกเป็นระดับชั้นต่างๆ เช่น ผู้รับผิดชอบของประเทศไทยคือ ns.thnic.net ผู้รับผิดชอบภายในสถาบันคือ crsc.kmitl.ac.th

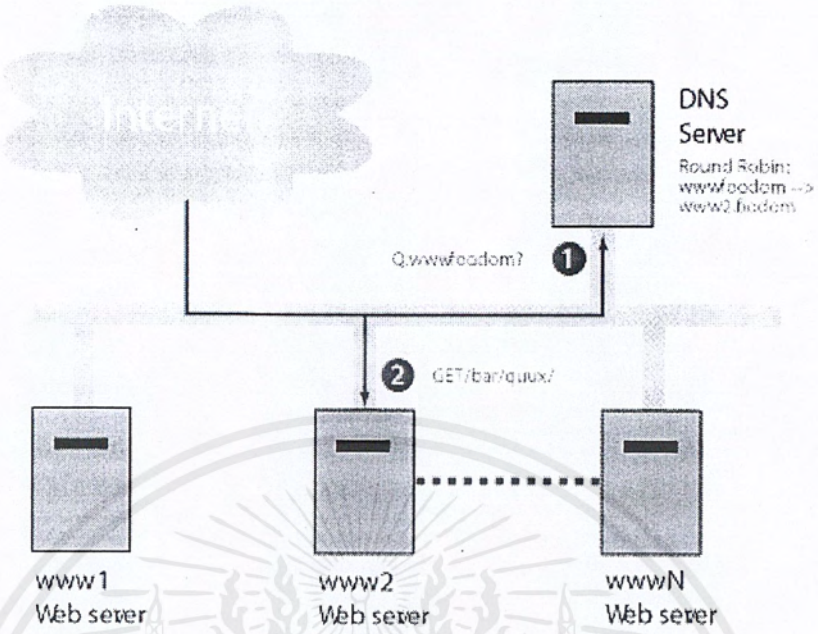
เครื่องที่ใช้เป็น Domain Name Server นั้น จะต้องมีทั้ง Primary Name Server และ Secondary Name Server โดยเมื่อใดก็ตามที่ Primary Name Server ไม่สามารถใช้งานได้ ก็ยังมีเครื่อง Secondary Name Server คอยทำหน้าที่ต่อไปได้

การแบ่งโหลดโดยใช้ Domain Name Server

การใช้งานในลักษณะนี้เป็นที่นิยมกันมากในปัจจุบัน เพราะสามารถทำได้ง่าย หลักการก็คือ จะทำการเก็บชื่อเพียงชื่อเดียวกับ IP Address หลายๆหมายเลข นั่นคือ จะทำการแม็ปชื่อเพียงชื่อเดียวไปยังหลายๆเครื่องนั่นเอง เช่น ชื่อ webcluster.ce.kmitl.ac.th ถูกแม็ปเข้ากับคอมพิวเตอร์ 3 เครื่อง มี IP Address ดังนี้ 161.246.5.176 , 161.246.5.177 , 161.246.5.178 เมื่อมีสัญญาณการร้องขอเข้ามาที่ Domain Name Server ก็จะทำการส่ง IP Address ของแต่ละเครื่องเรียงกันไปเรื่อยๆ ทำให้มีการแบ่งโหลดไปยังแต่ละเครื่องได้

ปัญหาของวิธีการนี้คือ แต่ละเครื่องอาจได้รับโหลดไม่เท่ากันได้ เพราะบางสัญญาณการร้องขออาจมีการใช้งานเครื่องเซิร์ฟเวอร์มาก ในขณะที่บางสัญญาณการร้องขอมีการใช้งานน้อย

รูปที่ 2.1 แสดงตัวอย่างของวิธีการนี้



รูปที่ 2.1 แสดงการใช้ Domain Name Server แบ่งโหลดให้เว็บเซิร์ฟเวอร์

2.3.2 การติดตั้ง Domain Name Server เพื่อแบ่งโหลดให้เว็บเซิร์ฟเวอร์

หากต้องการเป็น Domain Name Server ให้กับโดเมน ce.kmitl.ac.th ให้นำข้อความต่อจากนี้ไปต่อไว้ท้ายไฟล์ /etc/named.conf ของเครื่อง Primary Name Server

```
zone “ce.kmitl.ac.th” {
        type master;
        file “/var/named/master/ce.kmitl.ac.th”;
};
```

จากนั้นทำการสร้างไฟล์ /var/named/master/ce.kmitl.ac.th โดยมีเนื้อหาดังตัวอย่างนี้

```
@IN     SOA     icl20.ce.kmitl.ac.th root.icl20.ce.kmilt.ac.th.(
        991223162
        604800
        3600
        604800
        86400)
        IN NS  161.246.10.21.
        IN NS  161.246.10.22.

localhost       IN A    127.0.0.1

ftp             IN A    161.246.4.8
www             IN A    161.246.4.2

webcluster      IN A    161.246.5.176
                IN A    161.246.5.177
                IN A    161.246.5.178
```

2.4 ใช้ Reverse Proxy

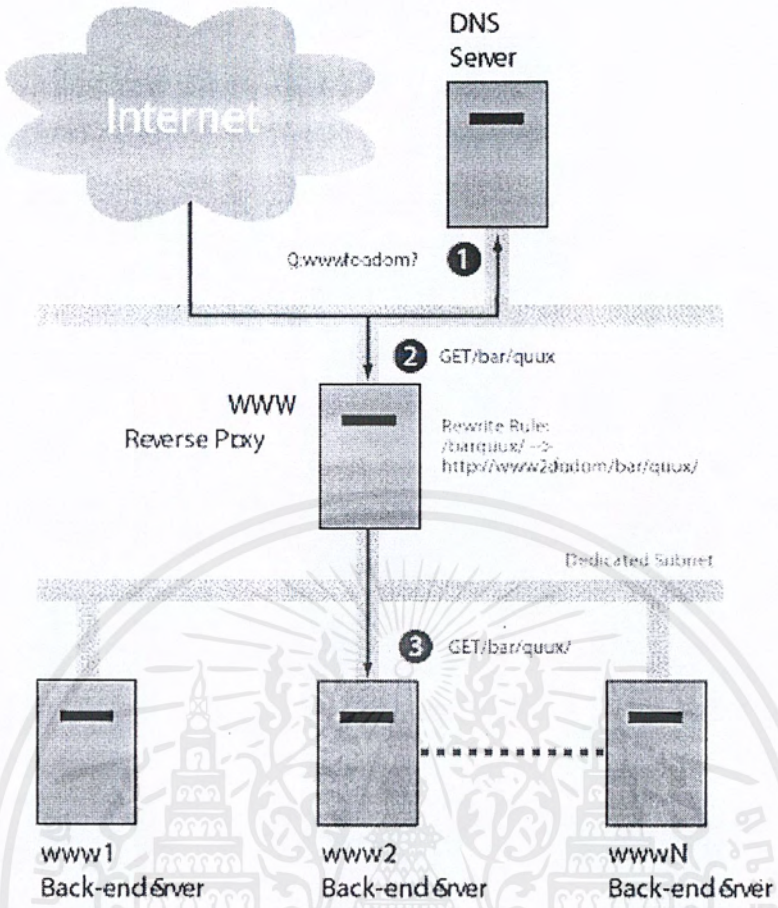
2.4.1 หลักการของ Reverse Proxy

ก่อนอื่นต้องทำความเข้าใจกับหลักการของ Forward Proxy Server กันก่อนโดยมันจะทำตัวเหมือนกับเป็นตัวแทนของไคลเอนต์ เมื่อไรก็ตามที่ไคลเอนต์มีการร้องขอไปยังเครื่องใดๆ ในอินเตอร์เน็ต ตัว Forward Proxy Server นี้ก็จะเป็นตัวที่ไปติดต่อกับเครื่องนั้นๆ เมื่อได้รับการตอบรับกลับมา ตัวมันก็จะส่งข้อมูลนั้นต่อไปให้กับไคลเอนต์ โดยมันอาจจะกระทำการใดๆกับข้อมูลนั้นก่อนหรือไม่ก็ได้ และข้อมูลนั้นๆก็จะถูกแคชเก็บไว้ที่ตัว Proxy และเมื่อใดก็ตามที่ข้อมูลที่ไคลเอนต์ร้องขอนั้น ได้มีอยู่แล้วภายในตัว Proxy มันก็จะทำการส่งต่อให้กับไคลเอนต์เลย ไม่ต้องไปร้องขอจากปลายทาง ทำให้ความเร็วในการตอบสนองเร็วขึ้นอย่างมาก แต่ข้อมูลที่แคชไว้นั้นจะอยู่ได้แค่ชั่วระยะเวลาหนึ่งเท่านั้น (Time To Live) ถ้าหากเกินช่วงระยะเวลานี้ ก็จะต้องไปร้องขอจากปลายทางเช่นเคย

สำหรับตัว Reverse Proxy Server ในทางกลับกัน จะเหมือนกับเป็นตัวแทนให้กับฝั่งเซิร์ฟเวอร์ เมื่อมีสัญญาณร้องขอเข้ามา ตัว Reverse Proxy Server จะเป็นตัวเลือกว่าจะเอาข้อมูลจากเว็บเซิร์ฟเวอร์เครื่องใด และถ้าข้อมูลที่ต้องการมีอยู่ในตัว Proxy อยู่แล้วก็ไม่จำเป็นที่จะต้องไปเอาข้อมูลมาจากตัวเว็บเซิร์ฟเวอร์ ข้อดีของกรณีนี้คือ เวลาที่ใช้ในการดึงข้อมูลออกจากตัว Proxy จะเร็วกว่าการดึงข้อมูลจากเว็บเซิร์ฟเวอร์โดยตรง

ยังมีอีกแนวทางหนึ่งในการประยุกต์ใช้ Reverse Proxy Server นั่นก็คือ กรณีที่เรามีเว็บเซิร์ฟเวอร์เพียงเครื่องเดียว แต่มี Reverse Proxy Server 2 เครื่อง แต่ละเครื่องชี้ไปที่เว็บเซิร์ฟเวอร์เครื่องเดียวกัน เมื่อถึงช่วงระยะเวลาหนึ่ง เมื่อ Proxy แต่ละตัวมีการแคชข้อมูลไว้มากๆ ก็จะเสมือนกับมีเว็บเซิร์ฟเวอร์ 2 เครื่องนั่นเอง และยังมีความเร็วในการดึงข้อมูลสูงกว่าอีกด้วย

รูปที่ 2.2 แสดงตัวอย่างของวิธีการนี้



รูปที่ 2.2 แสดงการใช้ Reverse Proxy Server แบ่งโหลดให้เว็บเซิร์ฟเวอร์

2.4.2 การติดตั้ง Reverse Proxy Server เพื่อแบ่งโหลดให้เว็บเซิร์ฟเวอร์

ในตัวอย่างนี้จะเป็นการใช้เว็บเซิร์ฟเวอร์เพียงเครื่องเดียว แต่มี Reverse Proxy Server 2 เครื่อง เพื่อทำการแคชข้อมูลจากเว็บเซิร์ฟเวอร์

ก่อนอื่นต้องทำการเพิ่มข้อความข้างล่างนี้ ที่ไฟล์ /var/named/master/ce.kmitl.ac.th ที่เครื่อง Domain Name Server เสียก่อน

```
webcluster      IN A    161.246.5.176
                IN A    161.246.5.177
```

จากนั้นก็ทำการคอนฟิกที่ไฟล์ path/to/squid/etc/squid.conf ที่เครื่อง Reverse Proxy Server ทั้งสองเครื่อง ดังนี้

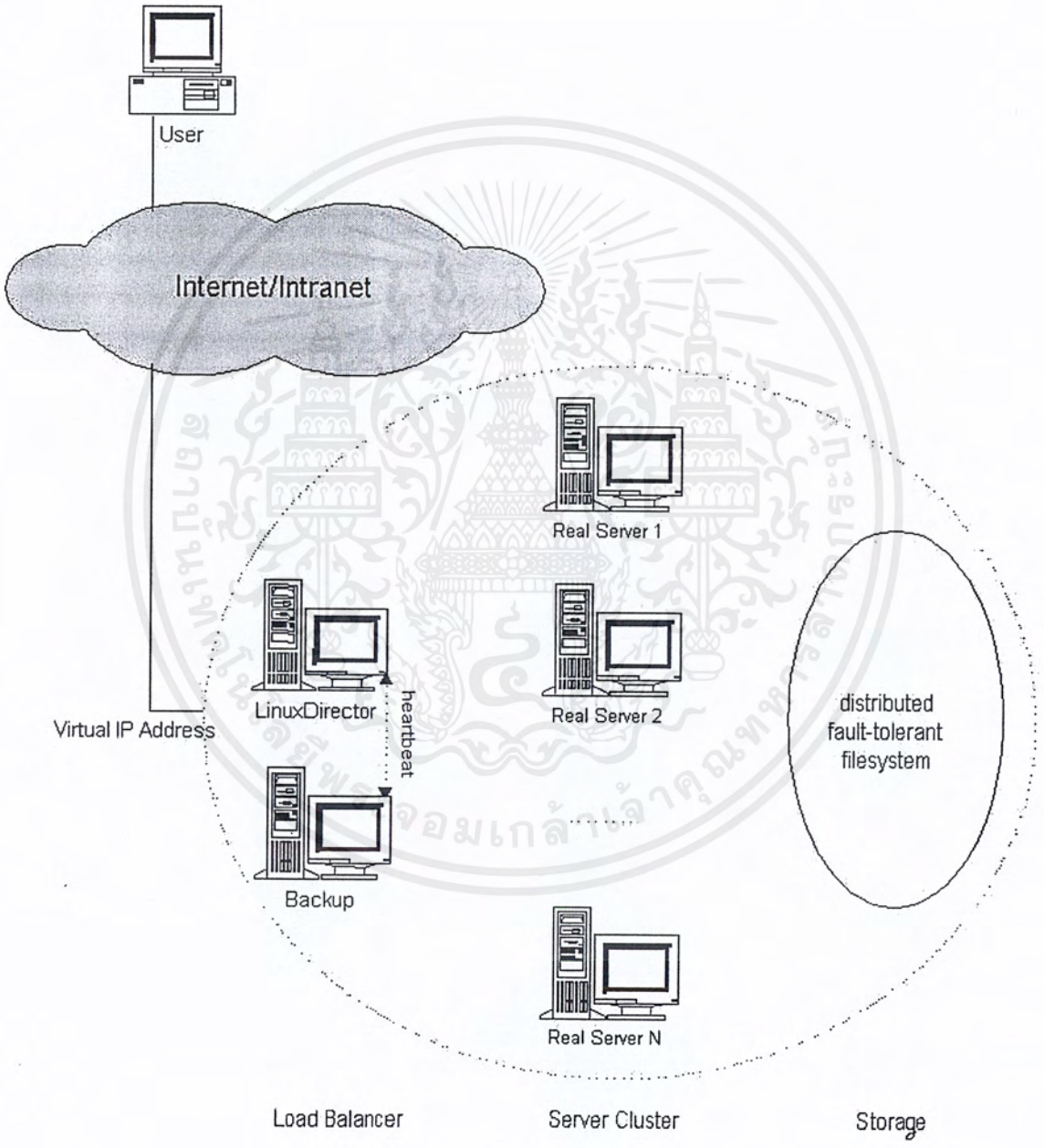
```
http_port 80

httpd_accel_host ชื่อของเว็บเซิร์ฟเวอร์

httpd_accel_port 80

httpd_accel_with_proxy on

httpd_accel_uses_host_header off
```

บทที่ 3

Linux Virtual Server

3.1 สถาปัตยกรรมโดยรวมของระบบ Linux Virtual Server

ระบบ Linux Virtual Server จะมีสถาปัตยกรรมโดยรวมของระบบเป็นไปตามรูปข้างล่างนี้



รูปที่ 3.1 แสดงสถาปัตยกรรมโดยรวมของระบบ Linux Virtual Server

จากรูปที่ 3.1 ระบบ Linux Virtual Server จะประกอบด้วย 3 ส่วนดังนี้คือ

- *Load Balancer* เป็นส่วนที่ติดต่อกับการร้องขอบริการทางอินเตอร์เน็ตจากไคลเอนต์ ที่มีเข้ามาในระบบ เครื่องโหลดบาลานซ์นี้จะมีหมายเลขประจำเครื่องที่สามารถติดต่อผ่านทางอินเตอร์เน็ตได้ และยังทำหน้าที่ในการส่งสัญญาณร้องขอบริการที่มีเข้ามาให้กับเครื่องเซิร์ฟเวอร์ต่างๆที่อยู่ภายในคลัสเตอร์ให้ทำงานตามสัญญาณร้องขอบริการที่มีเข้ามา
- *Server Pool* เป็นส่วนที่ประกอบด้วยเครื่องเซิร์ฟเวอร์ต่างๆที่อยู่ภายในคลัสเตอร์ที่ทำหน้าที่ในการทำงานกับสัญญาณร้องขอบริการทางอินเตอร์เน็ตที่เครื่องโหลดบาลานซ์ส่งมาให้ทำงาน
- *Backend Storage* เป็นส่วนที่ทำหน้าที่ในการเก็บข้อมูลโดยรวมของระบบ ซึ่งทำให้ระบบสามารถทำการจัดการกับข้อมูลที่ต้องให้บริการทางอินเตอร์เน็ตนั้นมีความเหมือนกันของเซิร์ฟเวอร์แต่ละเครื่องที่ต้องทำการบริการให้กับไคลเอนต์ที่ติดต่อเข้ามาผ่านทางเครื่องโหลดบาลานซ์

ขั้นตอนการทำงานของ Linux Virtual Server จะเป็นไปดังนี้คือ เครื่องโหลดบาลานซ์จะทำการจัดการกับสัญญาณการร้องขอบริการที่มีเข้ามาโดยการใช้เทคนิคทางด้าน IP Load Balancing ในการเลือกเครื่องเซิร์ฟเวอร์เครื่องหนึ่งจากเครื่องเซิร์ฟเวอร์ทั้งหลายที่มีอยู่ในคลัสเตอร์ให้ทำงานกับสัญญาณการร้องขอบริการที่มีเข้ามา และเครื่องโหลดบาลานซ์ยังทำการรักษาสถานะในการติดต่อกับเครื่องไคลเอนต์ และพร้อมกับทำการส่งต่อ packetไปให้กับเครื่องเซิร์ฟเวอร์ที่อยู่ภายในคลัสเตอร์ที่ได้ทำการเลือกไว้ ซึ่งงานทั้งหมดที่ทำที่เครื่องโหลดบาลานซ์นั้นจะทำภายใน kernel ของระบบปฏิบัติการ ดังนั้น overhead ของระบบนี้ในเครื่องโหลดบาลานซ์จะมีน้อย ส่งผลให้เครื่องโหลดบาลานซ์สามารถจัดการกับสัญญาณร้องขอที่มีเข้ามาได้มากกว่าเซิร์ฟเวอร์ทั่วไป ซึ่งจะทำให้ประสิทธิภาพในการทำงานโดยรวมของระบบดีขึ้นมาก โดยระบบสามารถรองรับกับสัญญาณการร้องขอบริการทางอินเตอร์เน็ตที่มากขึ้นได้อย่างมีประสิทธิภาพ

เครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์ที่ทำงานตามที่เครื่องโหลดบาลานซ์ส่งมาให้ทำนั้นจะทำให้เกิดความสามารถที่เรียกว่า Scalability โดยที่การ Scalability นั้นจะทำได้โดยการเพิ่มหรือลดเครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์ได้โดยตรงตามความต้องการใช้งาน เช่น เมื่อภาระงานที่ระบบต้องทำมีมากขึ้นเกินความสามารถของระบบที่จะรองรับได้ เราก็จะทำการเพิ่มเครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์เพื่อให้ระบบสามารถรองรับกับภาระงานที่มีมากขึ้นได้ ซึ่งประสิทธิภาพของระบบนั้นจะเพิ่มมากขึ้นตามจำนวนเครื่องเซิร์ฟเวอร์ที่มีอยู่ในคลัสเตอร์

ข้อได้เปรียบอีกประการหนึ่งของระบบคลัสเตอร์ก็คือมันสามารถทำการ Redundancy ได้กับทั้งฮาร์ดแวร์และซอร์ฟแวร์ ซึ่งทำให้มันมีความสามารถที่เรียกว่า Availability โดยความสามารถที่ว่านี้ก็คือระบบจะทำการตรวจจับความผิดพลาดที่แต่ละเครื่องเซิร์ฟเวอร์และเมื่อมันพบว่ามีความผิดพลาดเกิดขึ้นมันก็จะทำการปรับเปลี่ยนระบบได้อย่างเหมาะสมโดยอัตโนมัติ ซึ่งระบบจะทำการส่งภาระงานไปให้กับเครื่องเซิร์ฟเวอร์ที่เหลือภายในคลัสเตอร์ให้ทำงานแทนได้ โดยวิธีการนี้เราจะมีโปรแกรม monitor daemon ทำงานอยู่ที่เครื่องโหลดบาลานซ์ ซึ่งจะทำการตรวจสอบดูสภาพการทำงานของระบบคลัสเตอร์โดยจะตรวจสอบว่าเซิร์ฟเวอร์แต่ละเครื่องภายในคลัสเตอร์นั้นยังทำงานอยู่หรือไม่ ซึ่งถ้าเครื่องเซิร์ฟเวอร์เครื่องหนึ่งเครื่องใดไม่ทำการตอบรับกับสัญญาณ ICMP ping หรือไม่มีการตอบรับกับสัญญาณร้องขอบริการที่เครื่องโหลดบาลานซ์ส่งไปให้ทำงานภายในช่วงระยะเวลาหนึ่ง โปรแกรม monitor ที่เครื่องโหลดบาลานซ์ก็จะทำให้เซิร์ฟเวอร์เครื่องนั้นไม่ได้ถูกใช้

งานภายในระบบคลัสเตอร์ โดยการลบเซิร์ฟเวอร์เครื่องนั้นออกจาก scheduling table ที่อยู่ในเครื่องโหลดบาลานซ์ ซึ่งก็จะส่งผลให้เครื่องโหลดบาลานซ์ไม่ทำการส่งสัญญาณการร้องขอบริการที่มีเข้ามาให้กับเครื่องเซิร์ฟเวอร์ที่ทำงานผิดพลาดเครื่องนั้นภายในคลัสเตอร์ทำงานได้

เครื่องโหลดบาลานซ์อาจจะทำให้ทั้งระบบไม่สามารถทำงานได้เมื่อเครื่องโหลดบาลานซ์มีข้อผิดพลาดเกิดขึ้นและได้หยุดการทำงานหรือที่เรียกว่า Single Point of Failure เพื่อที่จะทำให้ระบบยังทำงานต่อไปได้แม้ว่าเครื่องโหลดบาลานซ์จะหยุดการทำงานอันเนื่องมาจากสาเหตุใดก็ตาม เราสามารถที่จะทำการแก้ไขได้โดยการติดตั้งเครื่องที่ทำหน้าที่แทนเครื่องโหลดบาลานซ์ที่ไม่สามารถทำงานได้โดยเราจะเรียกเครื่องนี้ว่าเครื่องสำรอง (backup) ซึ่งวิธีการก็คือเราจะใช้โปรแกรม daemon ที่ทำหน้าที่ในการตรวจสอบสถานะการทำงานของอีกเครื่องหนึ่งว่ายังคงทำงานอยู่หรือไม่ โดยจะให้โปรแกรม daemon นี้ทำงานอยู่ที่เครื่องที่ทำหน้าที่เป็นโหลดบาลานซ์ของระบบกับเครื่องที่ทำหน้าที่สำรอง ซึ่งโปรแกรม daemon นี้จะทำการตรวจสอบสถานะการทำงานผ่านทางสายแบบอนุกรม (serial line) หรือผ่านทาง UDP ในทุกๆช่วงเวลา เมื่อ daemon ที่เครื่องสำรองไม่ได้รับสัญญาณตอบรับจากเครื่องโหลดบาลานซ์ในช่วงเวลาที่กำหนดไว้ เครื่องสำรองก็จะทำการใช้วิธี ARP spoofing เพื่อทำให้เครื่องสำรองเปลี่ยนหมายเลขไอพีมาเป็นหมายเลขไอพีที่ให้บริการทางอินเตอร์เน็ตของเครื่องโหลดบาลานซ์ที่ทำหน้าที่รับสัญญาณการร้องขอบริการจากอินเตอร์เน็ต ต่อจากนั้นเครื่องสำรองก็จะทำหน้าที่แทนเครื่องโหลดบาลานซ์ และเมื่อเครื่องที่เป็นโหลดบาลานซ์ที่หยุดการทำงานไปกลับมาทำงานได้เหมือนเดิม การทำงานจะเป็นไปในสองวิธีนี้คือ

1. เครื่องที่เป็นโหลดบาลานซ์ที่กลับมาทำงานได้เหมือนเดิมจะกลายมาเป็นเครื่องสำรองให้กับเครื่องสำรองที่ทำหน้าที่แทนเครื่องโหลดบาลานซ์
2. เมื่อโปรแกรม daemon ที่เครื่องสำรองที่ทำหน้าที่แทนเครื่องโหลดบาลานซ์ได้รับสัญญาณตอบรับจากเครื่องที่เป็นโหลดบาลานซ์ที่กลับมาทำงานได้เหมือนเดิม เครื่องสำรองก็จะทำการคืนหมายเลขไอพีที่เป็นหมายเลขที่ใช้ในการบริการทางอินเตอร์เน็ตให้กับเครื่องโหลดบาลานซ์ที่กลับมาทำงานได้ใหม่

อย่างไรก็ตามทั้งสองวิธีนี้ในขั้นตอนที่เครื่องโหลดบาลานซ์เกิดทำงานผิดพลาดและเครื่องสำรองทำงานแทน จะทำให้การติดต่อของสัญญาณการร้องขอบริการก่อนหน้านั้นหายไป ซึ่งส่งผลให้ไคลเอนต์ต้องทำการส่งสัญญาณร้องขอบริการเข้ามาใหม่

จากรูปที่ 3.1 นั้นส่วนที่เป็น Backend Storage โดยปกติแล้วจะเป็นระบบไฟล์ที่คงทนต่อความผิดพลาด (distributed fault-tolerant file system) ตัวอย่างเช่น GFS, Coda หรือ Intermezzo ซึ่งระบบไฟล์เหล่านี้จะคำนึงถึงเรื่อง Availability กับ Scalability ของการเข้าถึงไฟล์ของระบบ ซึ่งเครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์จะทำการเข้าถึงระบบไฟล์เหมือนกับการเข้าถึงไฟล์ที่เครื่องของตนเอง อย่างไรก็ตามโปรแกรมประยุกต์ที่เป็นโปรแกรมเดียวกันจำนวนหลายโปรแกรมที่ทำงานภายในเซิร์ฟเวอร์แต่ละเครื่องภายในคลัสเตอร์อาจจะทำการเข้าถึงทรัพยากรของระบบเดียวกันแบบพร้อมกัน ซึ่งจะส่งผลให้เกิดความขัดแย้ง (conflict) ขึ้นภายในระบบ โดยปกติแล้วโปรแกรมประยุกต์แต่ละโปรแกรมจะต้องทำให้ไม่เกิดการขัดแย้งเกิดขึ้นเมื่อเข้าใช้งานทรัพยากรของระบบพร้อมกันเพื่อให้ทรัพยากรนั้นอยู่ในสถานะที่เสถียรซึ่งจำเป็นต้องมีโปรแกรมที่ทำหน้าที่นี้ (distributed lock manager) ซึ่งอาจจะอยู่ในระบบไฟล์ที่ใช้หรืออาจจะอยู่ภายนอกระบบไฟล์ที่ใช้ก็ได้

โปรแกรมจัดการนี้จะทำให้ผู้พัฒนาโปรแกรมประยุกต์สามารถที่จะทำการโปรแกรมได้อย่างง่ายกับการทำงานพร้อมกันกับการเข้าถึงทรัพยากรของระบบในโปรแกรมประยุกต์ที่ทำงานอยู่ในแต่ละเครื่องเซิร์ฟเวอร์ภายในระบบคลัสเตอร์

3.2 เทคนิคทางด้าน IP Load Balancing

เนื่องจากว่าเทคนิคทางด้าน IP Load Balancing นั้นจะมีความสามารถทางด้าน Scalability ที่ดี ซึ่งเทคนิคทางด้าน IP Load Balancing ที่ใช้ใน Linux Virtual Server นั้นจะมีอยู่ 3 วิธีด้วยกันคือ

1. Linux Virtual Server via NAT (Network Address Translation)
2. Linux Virtual Server via IP Tunneling
3. Linux Virtual Server via Direct Routing

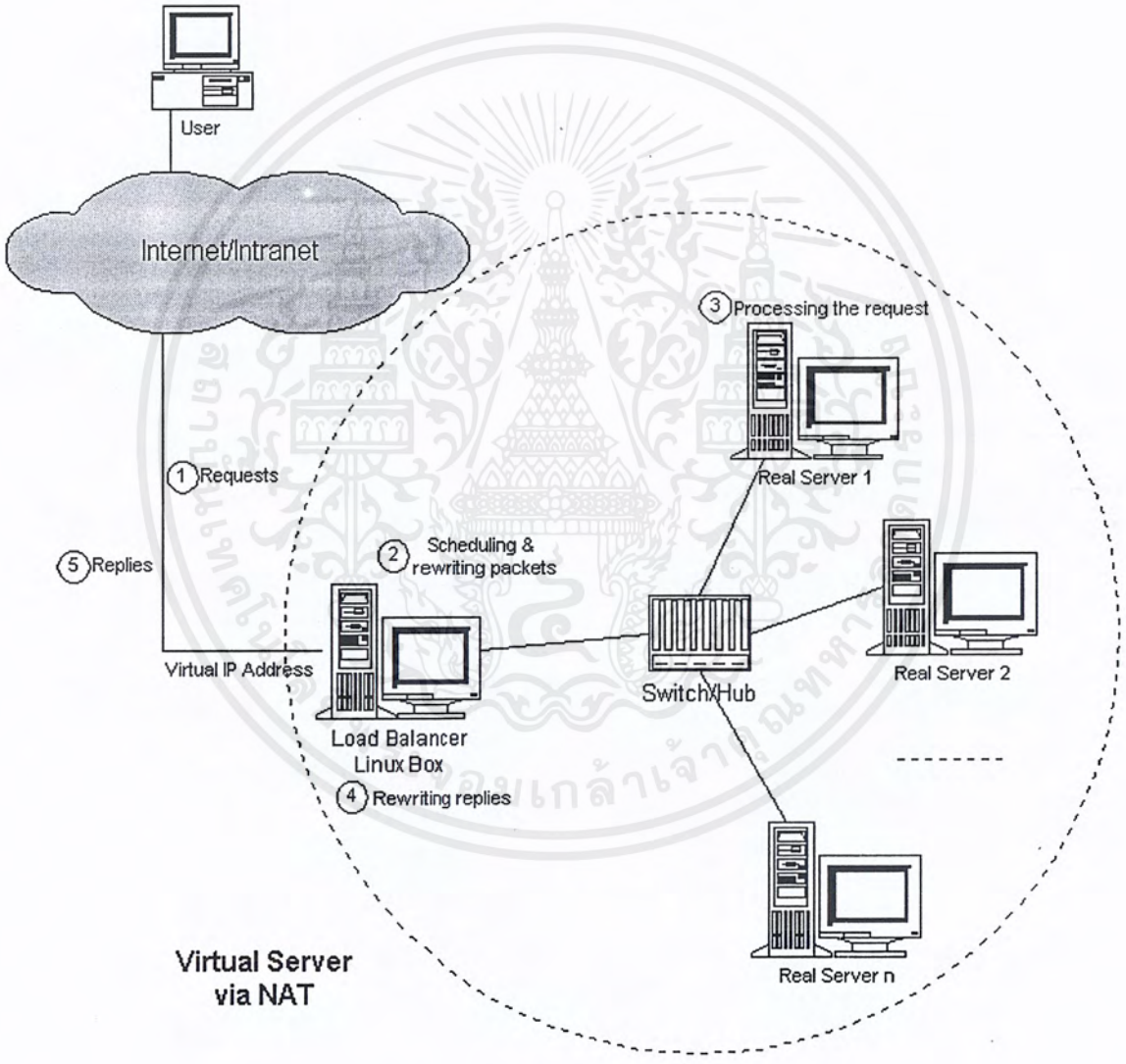
ในการติดตั้งระบบนี้เครื่องที่ทำหน้าที่เป็นโหลดบาลานซ์จะต้องทำการเพิ่มวิธีการเหล่านี้ (patch) ไปใน kernel ของระบบปฏิบัติการลีนุกซ์ ซึ่งจะทำให้เครื่องโหลดบาลานซ์ทำหน้าที่ในการติดต่อกับสัญญาณการร้องขอบริการของเครื่องไคลเอนต์ โดยที่เครื่องไคลเอนต์ จะรู้เพียงหมายเลขประจำเครื่องของเครื่องโหลดบาลานซ์ที่ใช้เป็นหมายเลขไอพีที่ให้บริการทางอินเตอร์เน็ต จากนั้นเครื่องโหลดบาลานซ์จะทำการส่งสัญญาณร้องขอที่มีเข้ามาให้กับเครื่องเซิร์ฟเวอร์เครื่องใดเครื่องหนึ่งภายในคลัสเตอร์ ซึ่งเซิร์ฟเวอร์เครื่องนั้นจะทำงานกับสัญญาณร้องขอที่มีเข้ามา โดยปกติแล้วเครื่องเซิร์ฟเวอร์ที่อยู่ภายในคลัสเตอร์จะมีรูปแบบและการทำงานที่เหมือนกันหมด และจะมีการบริการทางอินเตอร์เน็ตที่เหมือนกัน โดยที่ข้อมูลที่ใช้ในการบริการทางอินเตอร์เน็ตนั้นอาจจะอยู่ในแต่ละเครื่องเซิร์ฟเวอร์เอง หรือจะใช้ข้อมูลร่วมกันผ่านทางเครือข่าย หรือจะทำโดยการมีระบบไฟล์ที่ใช้ร่วมกันของระบบ และต่อจากนี้ไปจะเป็นการกล่าวถึงหลักการทำงานของเทคนิคทางด้าน IP Load Balancing ทั้ง 3 วิธีพร้อมทั้งเปรียบเทียบถึงข้อได้เปรียบและข้อเสียเปรียบของแต่ละวิธี

3.2.1 Linux Virtual Server via NAT (Network Address Translation)

เนื่องมาจากความขาดแคลนหมายเลขไอพีและปัญหาด้านความปลอดภัยใน IPv4 เครือข่ายต่างๆ จำนวนมากได้หันมาใช้หมายเลขไอพีภายในที่ไม่สามารถใช้งานได้ทางอินเตอร์เน็ต เช่น 10.0.0.0/255.0.0.0 เป็นต้น หลักการ Network Address Translation จึงเกิดขึ้นเพื่อให้เครื่องในเครือข่ายภายในสามารถติดต่อออกทางอินเตอร์เน็ตได้ และสามารถเข้าถึงได้จากเครื่องที่อยู่ในอินเตอร์เน็ต หลักการของ NAT จะอาศัยความจริงที่ว่า header ของ packet สามารถที่จะทำการเปลี่ยนแปลงอย่างเหมาะสมตามความต้องการใช้งาน ซึ่งจะส่งผลให้ไคลเอนต์เชื่อว่าตนเองนั้นกำลังทำการติดต่ออยู่กับหมายเลขประจำเครื่องของเครื่องที่ให้บริการทางอินเตอร์เน็ตเพียงหมายเลขเดียว และเครื่องเซิร์ฟเวอร์ต่างๆที่อยู่ภายในคลัสเตอร์นั้นแต่ละเครื่องก็จะมีหมายเลขประจำเครื่องของมันเองซึ่งแต่ละเครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์จะเชื่อว่าตนเองกำลังทำการติดต่อกับไคลเอนต์โดยตรง ซึ่งลักษณะรูปแบบนี้สามารถที่จะนำมาสร้าง Virtual Server ตัวอย่างเช่น การบริการแบบขนานที่มีเครื่องเซิร์ฟเวอร์หลายเครื่องที่มีหมายเลขประจำเครื่องเป็นของตนเองสามารถที่จะนำมาทำเป็นการบริการแบบ Virtual Service ที่ทำงานโดยผ่านหมายเลขไอพีเพียงหมายเลขเดียวได้

วิธีการทำงานของ Network Address Translation จะเป็นการจับคู่หมายเลขไอพีจากกลุ่มหนึ่งไปยังหมายเลขไอพีอีกกลุ่มหนึ่ง เมื่อการจับคู่เป็นแบบ N-to-N วิธีการนี้จะเรียกว่า Static Network Address Translation และถ้าการจับคู่เป็นแบบ M-to-N (M>N) จะเรียกวิธีการนี้ว่า Dynamic Address Translation และหลักการของ Network Address Port Translation จะมีวิธีการมาจากพื้นฐานของ NAT ซึ่งหมายเลขไอพีจำนวนหนึ่งและหลายหมายเลข port ใน TCP/UDP จะถูกแปลงเป็นหมายเลขไอพีเดียวและหลายหมายเลข port วิธีการแบบนี้จะเป็นการจับคู่แบบ N-to-1 ซึ่งจะเป็นหลักการที่ Linux IP Masquerading นำมาใช้

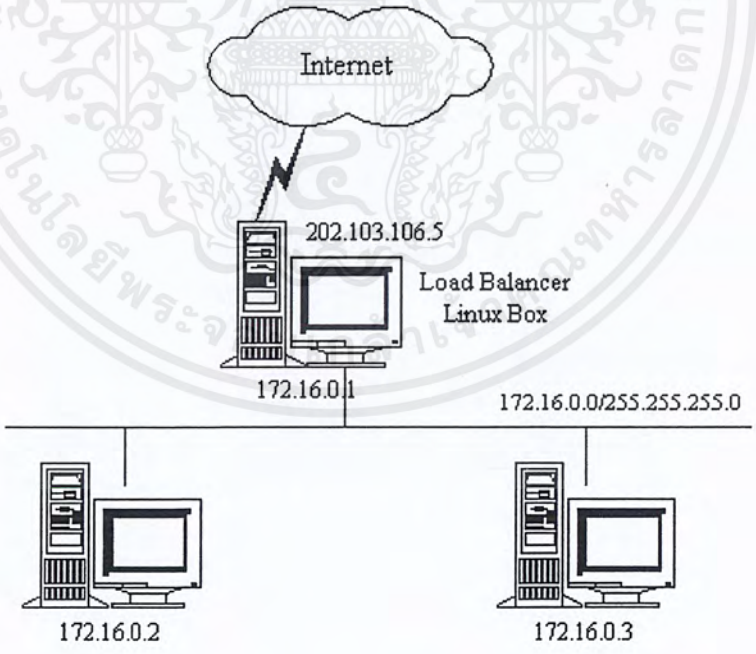
Virtual Server via NAT จะใช้หลักการของ Network Address Port Translation ในการทำงาน ซึ่งจะอาศัยจากการทำงานของ Linux IP Masquerading ที่มีอยู่ในลีนุกซ์



รูปที่ 3.2 แสดงสถาปัตยกรรมของระบบ Linux Virtual Server via NAT

สถาปัตยกรรมของระบบ Linux Virtual Server via NAT จะแสดงให้เห็นตามรูปที่ 3.2 โดยที่เครื่องโหลดบาลานซ์และเครื่องเซิร์ฟเวอร์ต่างๆที่อยู่ภายในคลัสเตอร์จะทำการต่อกันโดยใช้ switch หรือ hub ซึ่งขั้นตอนการทำงานของ Virtual Server via NAT จะทำงานเป็นไปดังนี้ เมื่อผู้ใช้ได้ทำการส่งสัญญาณร้องขอบริการมาที่ระบบคลัสเตอร์ที่ทำการบริการแบบ virtual service ซึ่งสัญญาณร้องขอที่มีเข้ามานั้นจะมีที่อยู่ปลายทางเป็นหมายเลขไอพีที่ทำการบริการทางอินเตอร์เน็ต (virtual IP address) ซึ่งสัญญาณร้องขอนี้จะเข้ามาที่ระบบคลัสเตอร์โดยผ่านเครื่องโหลดบาลานซ์ ต่อจากนั้นเครื่องโหลดบาลานซ์จะทำการตรวจสอบดูว่าหมายเลขไอพีและหมายเลข port ว่าเป็นของ virtual IP address หรือไม่ ถ้าใช่เครื่องโหลดบาลานซ์จะทำการเลือกเซิร์ฟเวอร์ที่จะทำการประมวลผลภายในคลัสเตอร์โดยวิธีการเลือกนั้นจะเป็นไปตาม Scheduling Algorithm ที่ได้เลือกไว้และการติดต่อจะถูกเก็บไว้ใน hash table ซึ่งจะใช้สำหรับบันทึกการติดต่อที่มีเข้ามา จากนั้นหมายเลขไอพีปลายทางกับหมายเลข port ของ packet ที่มีเข้ามาจะถูกเขียนขึ้นใหม่ให้เป็นหมายเลขไอพีและหมายเลข port ของเครื่องเซิร์ฟเวอร์ที่ได้ถูกเลือกให้ทำงานกับสัญญาณร้องขอที่มีเข้ามานั้น และจากนั้นก็จะทำการส่ง packet ไปให้กับเซิร์ฟเวอร์เครื่องนั้น และเมื่อการติดต่อที่มีเข้ามาใหม่เป็นการติดต่อที่ได้สร้างขึ้นแล้ว ซึ่งได้ทำการบันทึกอยู่ใน hash table การทำงานก็จะทำโดยการเขียน header ของ packet ใหม่และทำการส่งไปให้กับเครื่องเซิร์ฟเวอร์ที่ได้ทำการบันทึกใน hash table และเมื่อ packet ตอบรับได้ทำการส่งกลับมาให้กับเครื่องโหลดบาลานซ์ ซึ่งเครื่องโหลดบาลานซ์ก็จะทำการเขียนหมายเลขไอพีและหมายเลข port ต้นทางใหม่ให้เป็นหมายเลขของ Virtual IP Address เมื่อการติดต่อนั้นจบการทำงานหรือเกิดการ timeout สิ่งที่ได้บันทึกใน hash table ก็จะถูกลบทิ้งไป

ตัวอย่างที่แสดงการทำงานโดยละเอียดของวิธีการแบบ NAT จะเป็นไปตามการติดตั้งในรูปที่ 3.3



รูปที่ 3.3 แสดงตัวอย่างการติดตั้งของระบบ Linux Virtual Server via NAT

จากรูปที่ 3.3 เซิร์ฟเวอร์ภายในระบบคลัสเตอร์สามารถที่จะทำงานภายใต้ระบบปฏิบัติการอะไรก็ได้ แต่ระบบปฏิบัติการนั้นจำเป็นต้องทำการสนับสนุนกับโพรโตคอล TCP/IP และเครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์จะต้องทำการติดตั้งค่า default route ให้เป็นหมายเลขไอพีของเครื่องโหลดบาลานซ์ ในที่นี้จะเป็นหมายเลขไอพี 172.16.0.1 โดยที่โปรแกรม ipfwadm จะถูกใช้ในเครื่องโหลดบาลานซ์เพื่อที่จะทำให้เครื่องโหลดบาลานซ์สามารถที่จะทำการรับ packet จากเครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์ได้ ตามตัวอย่างในรูปที่ 3.3 คำสั่งที่ใช้จะเป็น

```
echo 1 > /proc/sys/net/ipv4/ip_forward
ipfwadm –F –a m –S 172.16.0.0/24 –D 0.0.0.0/0
```

ตารางต่อไปนี้จะแสดงกฎที่กำหนดไว้ในเครื่องที่ทำหน้าที่เป็นเครื่องโหลดบาลานซ์

Protocol	Virtual IP Address	Port	Real IP Address	Port	Weight
TCP	202.103.106.5	80	172.16.0.2	80	1
			172.16.0.3	8000	2

ตารางที่ 3.1 แสดงตัวอย่างกฎที่กำหนดการทำงานของ Virtual Server via NAT

สัญญาณการร้องขอบริการทุกสัญญาณที่มีหมายเลขไอพีปลายทางเป็น 202.103.106.5 และทำงานที่ port หมายเลข 80 จะถูกเครื่องโหลดบาลานซ์ทำการส่งสัญญาณการร้องขอบริการที่มีเข้ามาไปยังเครื่องเซิร์ฟเวอร์ที่อยู่ภายในคลัสเตอร์ที่มีหมายเลขไอพีเป็น 172.16.0.2 ทำงานที่ port หมายเลข 80 กับเครื่องเซิร์ฟเวอร์ที่มีหมายเลขไอพีเป็น 172.106.0.3 ทำงานที่ port หมายเลข 8000

ขั้นตอนการทำงานของ Packet rewriting จะทำงานดังนี้คือ เมื่อ packet การร้องขอบริการเข้ามาที่เครื่องโหลดบาลานซ์ ซึ่ง packet นั้นจะมีหมายเลขไอพีของเครื่องต้นทางกับเครื่องปลายทางเป็นดังนี้

SOURCE 202.100.1.2:3456 DESTINATION 202.103.106.5:80

ต่อจากนั้นเครื่องโหลดบาลานซ์จะทำการเลือกเครื่องเซิร์ฟเวอร์ที่อยู่ภายในคลัสเตอร์ตาม scheduling ที่ใช้ ซึ่งสมมุติว่าเป็น 172.16.0.3:8000 โดยที่ packet นั้นจะโดนเครื่องโหลดบาลานซ์ทำการเขียน packet ใหม่และส่งไปให้กับเครื่องเซิร์ฟเวอร์เครื่องนั้น ซึ่ง packet ที่เขียนใหม่จะมีค่าเป็นดังนี้

SOURCE 202.100.1.2:3456 DESTINATION 172.16.0.3:8000

จากนั้นสัญญาณตอบรับที่ส่งมาให้กับเครื่องโหลดบาลานซ์จะมีค่าเป็นดังนี้

SOURCE 172.16.0.3:8000 DESTINATION 202.100.1.2:3456

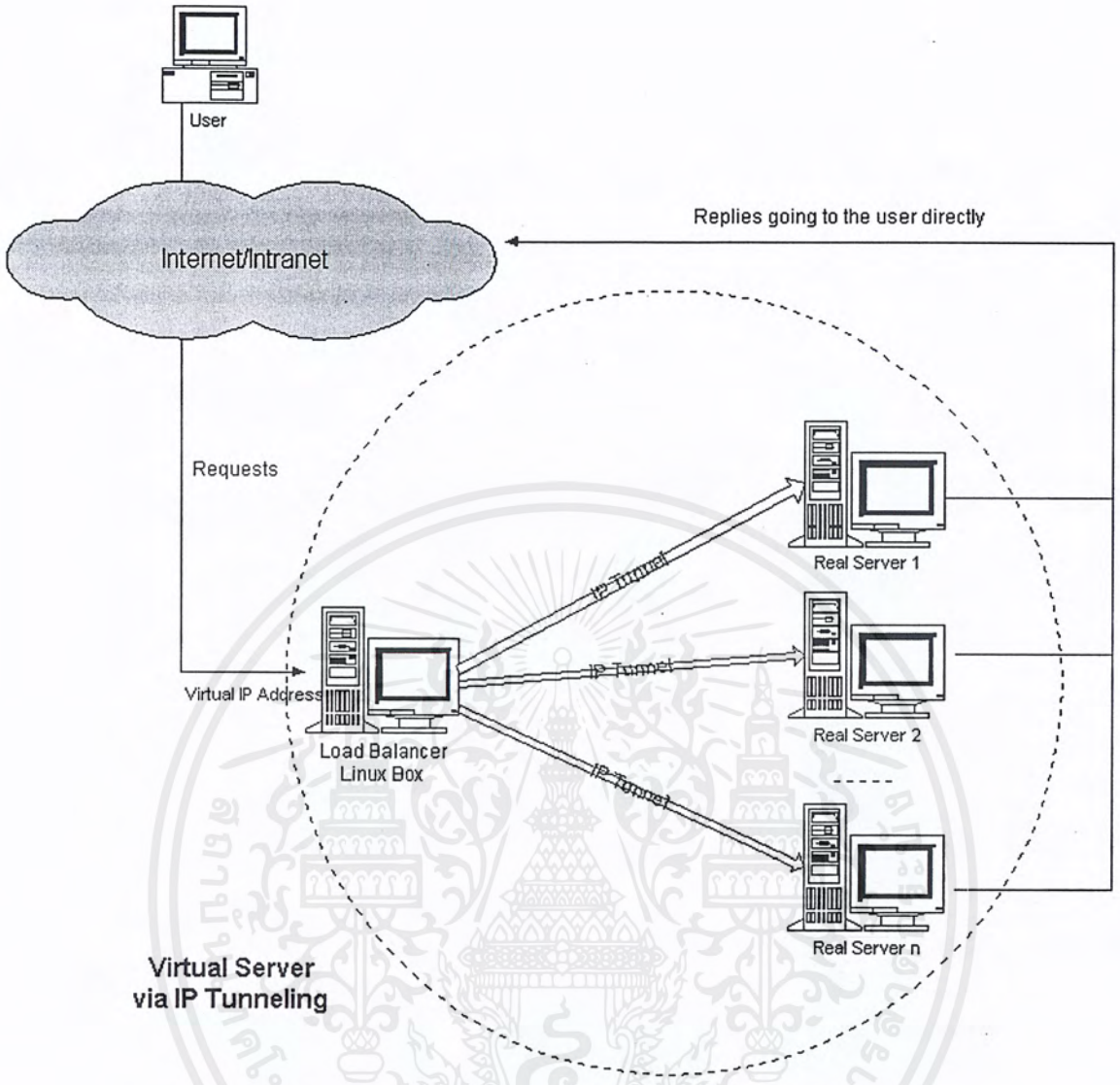
ต่อจากนั้นเครื่องโหลดบาลานซ์จะทำการเขียน packet ใหม่และส่งไปให้กับเครื่อง ไคลเอนต์ ซึ่ง packet ที่เขียนใหม่จะมีค่าเป็นดังนี้

SOURCE 202.103.106.5:80 DESTINATION 202.100.1.2:3456

3.2.2 Linux Virtual Server via IP Tunneling

วิธีการของ IP Tunneling เป็นเทคนิคที่ทำการ encapsulate กับ IP datagram ภายใน IP datagram อีกทีหนึ่ง ซึ่งจะทำให้ datagram ที่มีที่อยู่ปลายทางเป็นหมายเลขไอพีหนึ่งจะถูกทำการ encapsulate เป็นหมายเลขไอพีอีกหมายเลขหนึ่งและทำการ redirect ไปให้กับหมายเลขไอพีที่ทำการ encapsulate นั้น ซึ่งเทคนิคนี้สามารถนำมาใช้สร้าง Virtual Server ซึ่งเครื่องที่เป็นโหลดบาลานซ์จะทำการ tunnel กับสัญญาณร้องขอที่มีเข้ามาไปให้กับเซิร์ฟเวอร์เครื่องต่างๆภายในคลัสเตอร์ และเครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์นั้นจะทำงานกับสัญญาณการร้องขอบริการที่เครื่องโหลดบาลานซ์ส่งมาให้ จากนั้นก็จะทำการส่งผลลัพธ์ไปให้กับไคลเอนต์โดยตรง ซึ่งจะทำให้บริการที่ระบบให้กับไคลเอนต์ เป็นแบบ Virtual Server ที่ใช้หมายเลขไอพีที่ให้บริการทางอินเตอร์เน็ตเพียงหมายเลขเดียว

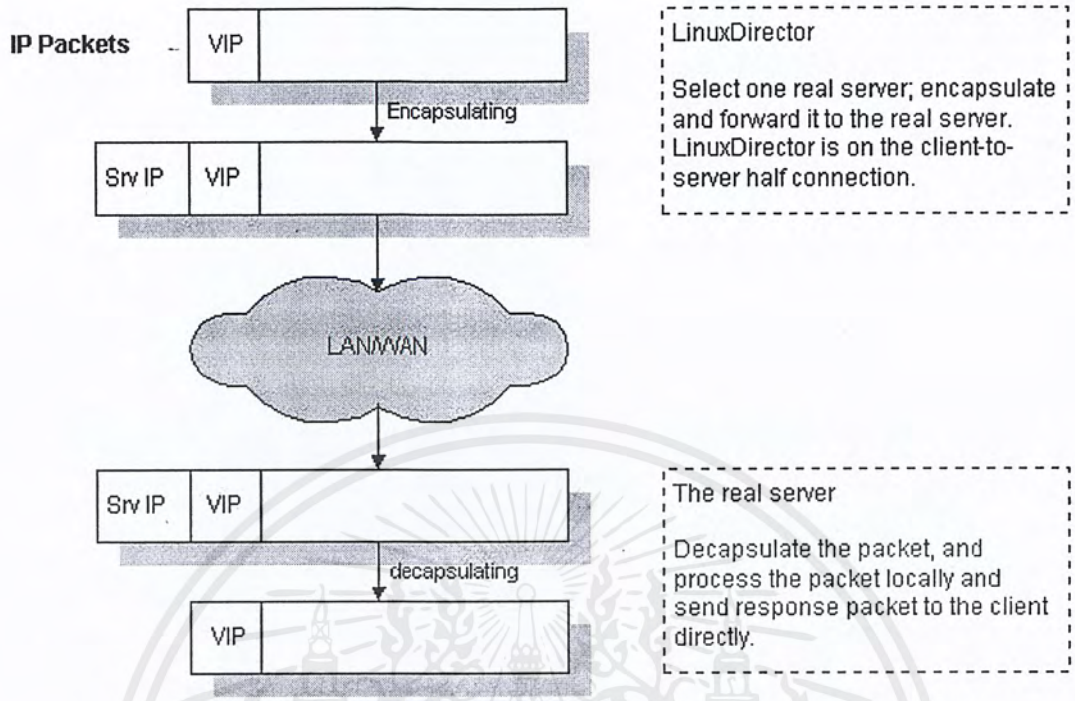
สถาปัตยกรรมของ Virtual Server via IP Tunneling จะถูกแสดงอยู่ในรูปที่ 3.4 โดยเครื่องที่เป็นเซิร์ฟเวอร์ภายในคลัสเตอร์นั้นสามารถมีหมายเลขไอพีจริงที่สามารถใช้ได้ในอินเตอร์เน็ต ซึ่งสามารถอยู่ที่เครือข่ายไหนก็ได้สักแห่ง แต่ว่าเครื่องเซิร์ฟเวอร์เหล่านั้นจะต้องทำการสนับสนุนกับโพรโตคอล IP Tunneling และที่เครื่องเซิร์ฟเวอร์เหล่านั้นจำเป็นที่จะต้องมีอินเตอร์เฟสที่ทำการติดตั้งเป็น Virtual IP หรือหมายเลขไอพีที่ใช้ในการบริการทางอินเตอร์เน็ต



รูปที่ 3.4 แสดงสถาปัตยกรรมของระบบ Linux Virtual Server via IP Tunneling

ขั้นตอนการทำงานของ Linux Virtual Server via IP Tunneling นั้นจะเหมือนกันกับ Linux Virtual Server via NAT แต่จะมีข้อแตกต่างตรงที่ว่าวิธีการของ Linux Virtual Server via IP Tunneling นั้นเครื่องที่เป็นโหลดบาลานซ์เมื่อได้รับสัญญาณการร้องขอบริการทางอินเตอร์เน็ตก็จะทำการ encapsulate กับ IP datagram ภายใน IP datagram อีกทีหนึ่งของ packet ที่มีเข้ามา และก็จะทำการส่ง packet นั้นไปให้กับเครื่องเซิร์ฟเวอร์ที่ได้ทำการเลือก เมื่อเครื่องเซิร์ฟเวอร์เครื่องที่ได้เลือกนั้นได้รับ packet ที่ผ่านการ encapsulate โดยเครื่องโหลดบาลานซ์ เครื่องเซิร์ฟเวอร์นั้นก็จะทำการ decapsulate กับ packet นั้นและพบว่าหมายเลขไอพีปลายทางที่จะส่งนั้นเป็นของ Virtual IP ซึ่งก็จะตรงกับอินเตอร์เฟสแบบ tunnel ที่เครื่องเซิร์ฟเวอร์นั้นได้ทำการติดตั้งเป็น Virtual IP ส่งผลให้เครื่องเซิร์ฟเวอร์เครื่องนั้นทำงานกับการร้องขอบริการและส่งผลลัพธ์ไปให้กับ ไคลเอนต์โดยตรง

ขั้นตอนการทำงานของวิธีการ Linux Virtual Server via IP Tunneling จะเป็นไปตามรูปที่ 3.5



รูปที่ 3.5 แสดงขั้นตอนการทำงานของ Linux Virtual Server via IP Tunneling

จากรูปที่ 3.5 เครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์ของระบบนี้สามารถที่จะมีหมายเลขไอพีจริงที่สามารถใช้ได้ในอินเตอร์เน็ตที่จะอยู่ในเครือข่ายไหนก็ได้ ซึ่งจะทำให้ระบบนี้มีการกระจายไปตามพื้นที่บริการต่างๆที่อยู่ห่างไกลกัน แต่เครื่องเซิร์ฟเวอร์ที่ใช้ในระบบนี้จำเป็นต้องสนับสนุนกับโพรโตคอลแบบ IP encapsulate โดยที่เครื่องเซิร์ฟเวอร์ต่างๆที่อยู่ภายในคลัสเตอร์ทุกเครื่องจะทำการติดตั้งอุปกรณ์อินเตอร์เฟสที่เป็นแบบ Tunnel ซึ่งจะทำให้เครื่องเซิร์ฟเวอร์ทุกเครื่องในคลัสเตอร์ทำการ decapsulate กับ packet ที่ถูก encapsulate มาโดยเครื่องโหลดบาลานซ์ และหมายเลขไอพีที่เป็น Virtual Address จะต้องทำการติดตั้งในอุปกรณ์อินเตอร์เฟสที่ไม่ทำการ ARP response

ในการทำงานนั้นเมื่อ packet ที่ถูก encapsulate มาถึงเค่รื่องเซิร์ฟเวอร์ภายในคลัสเตอร์ ซึ่งเครื่องเซิร์ฟเวอร์นี้ก็จะทำการ decapsulate และพบว่า packet นั้นส่งมาให้กับที่อยู่ปลายทางเป็น Virtual IP Address และมันก็จะพบว่าเป็นของที่เครื่องมันเอง ต่อจากนั้นมันก็จะทำงานกับ request นั้นและทำการส่งผลลัพธ์ไปให้กับ ไคลเอนต์ โดยตรง

ตัวอย่างวิธีการใช้งานจะเป็นดังนี้ ตารางที่ 3.2 จะแสดงตัวอย่างกฎที่ได้ทำการติดตั้งในเครื่องโหลดบาลานซ์โดยวิธีการ IP Tunneling โดยที่ port ที่ทำงานกับสัญญาณการร้องขอบริการของเซิร์ฟเวอร์ภายในคลัสเตอร์จะต้องเป็น port เดียวกับเครื่องโหลดบาลานซ์

Protocol	Virtual IP Address	Port	Real IP Address	Weight
TCP	202.103.106.5	80	202.103.107.2	1
			202.103.106.3	2

ตารางที่ 3.2 แสดงตัวอย่างกฎที่เป็นการติดตั้งแบบ Linux Virtual Server via IP Tunneling

จากตารางที่ 3.2 สัญญาณการร้องขอบริการทุกสัญญาณที่เข้ามาทางเครื่องโหลดบาลานซ์ที่มีหมายเลขไอพีเป็น 202.103.106.5 ที่ port หมายเลข 80 ซึ่งเครื่องโหลดบาลานซ์จะทำการกระจายสัญญาณการร้องขอบริการไปให้กับเครื่องเซิร์ฟเวอร์ที่มีหมายเลขไอพีเป็น 202.103.107.2 ที่ port หมายเลข 80 และที่เครื่องเซิร์ฟเวอร์ที่มีหมายเลขไอพีเป็น 202.103.106.3 ที่ port หมายเลข 80

เราสามารถที่จะใช้คำสั่งต่อไปนี้เพื่อที่จะทำให้ระบบทำงานตามในตารางที่ 3.2

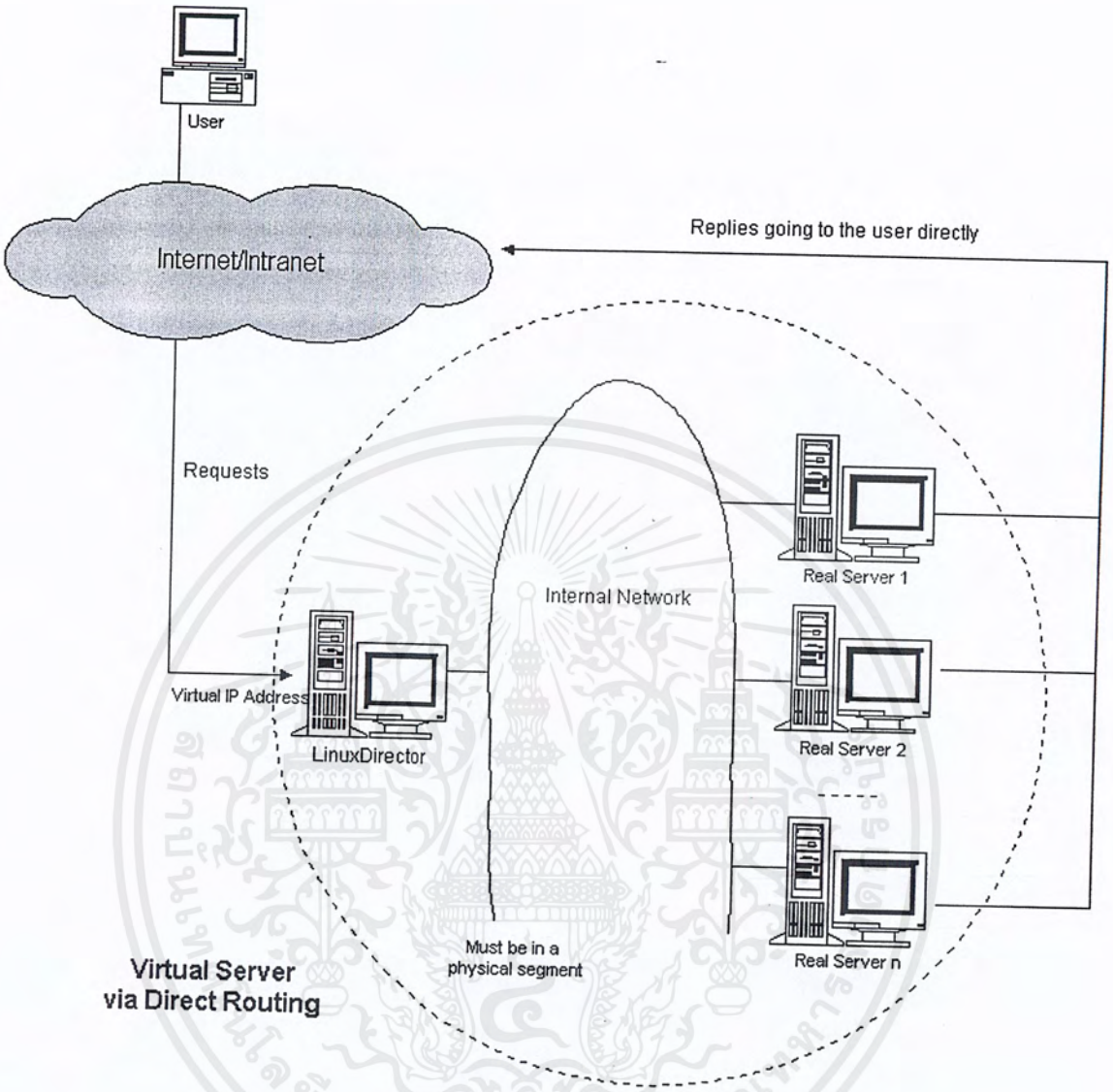
1. สำหรับ kernel 2.0.x

```
ippfvsadm –A –t 202.103.106.5:80 –R 202.103.107.2 –w 1
ippfvsadm –A –t 202.103.106.5:80 –R 202.103.106.3 –w 2
```

2. สำหรับ kernel 2.2.x

```
ipvsadm –A –t 202.103.106.5:80 –s wlc
ipvsadm –a –t 202.103.106.5:80 –R 202.103.107.2 –i –w 1
ipvsadm –a -t 202.103.106.5:80 –R 202.103.106.3 –i –w 2
```

3.2.3 Linux Virtual Server via Direct Routing

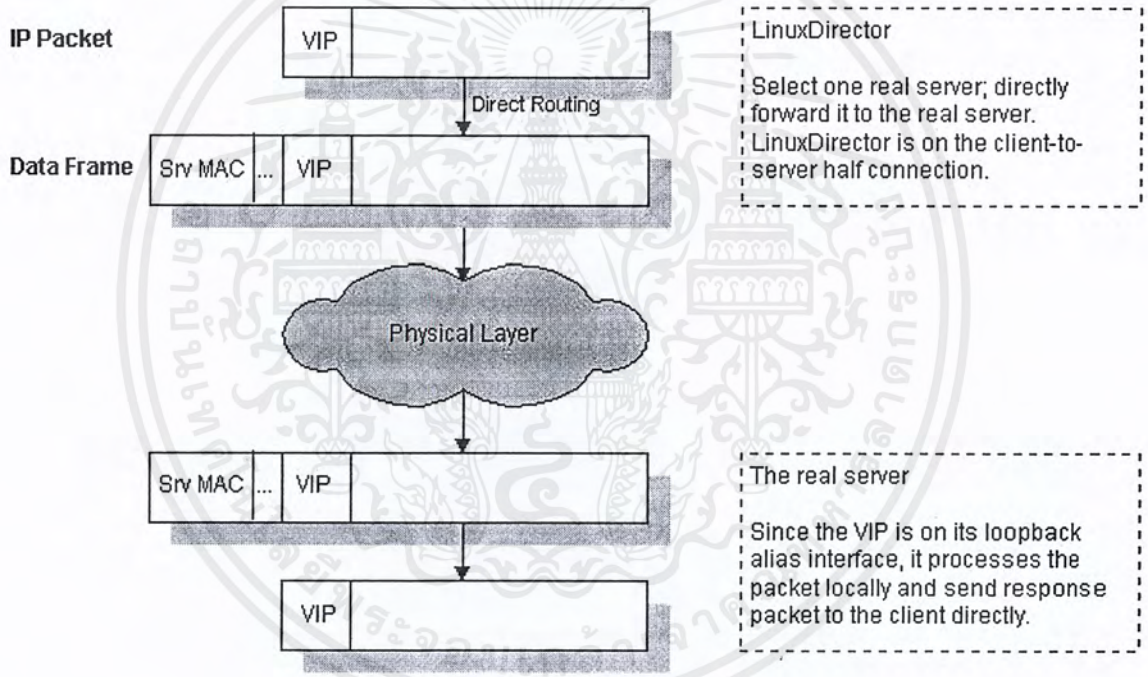


รูปที่ 3.6 แสดงสถาปัตยกรรมของระบบ Linux Virtual Server via Direct Routing

เทคนิคทางด้าน IP Load Balancing วิธีนี้จะคล้ายกันกับวิธีการที่ใช้สร้าง Net Dispatcher ของบริษัท IBM สถาปัตยกรรมของระบบ Linux Virtual Server via Direct Routing จะแสดงอยู่ในรูปที่ 3.6 โดยเครื่องที่ทำหน้าที่เป็นโหลดบาลานซ์และเครื่องเซิร์ฟเวอร์ต่างๆที่อยู่ภายในคลัสเตอร์จะทำการต่อกันใน LAN วงเดียวกัน เช่น ต่อกันโดย Hub หรือโดย Switch โดยที่เครื่องโหลดบาลานซ์และเครื่องเซิร์ฟเวอร์ต่างๆภายในคลัสเตอร์จะใช้หมายเลขไอพีที่เป็น Virtual IP ร่วมกัน โดยที่เซิร์ฟเวอร์ทุกเครื่องที่อยู่ภายในคลัสเตอร์จะมีการใช้อินเตอร์เฟสเสมือนแบบ loopback ที่ทำการติดตั้งเป็น Virtual IP และเครื่องโหลดบาลานซ์จะมีการใช้อินเตอร์เฟสที่ทำการติดตั้งเป็น Virtual IP เพื่อทำการรับกับสัญญาณการร้องขอบริการที่มีเข้ามาในระบบคลัสเตอร์

ขั้นตอนการทำงานของระบบคลัสเตอร์แบบ Linux Virutal Server via Direct Routing นั้นจะทำงานเหมือนกับ Linux Virtual Server via NAT กับ Linux Virtual Server via IP Tunneling โดยความแตกต่างจะอยู่ที่เครื่องโหลดบาลานซ์เมื่อได้รับสัญญาณการร้องขอบริการทางอินเตอร์เน็ตแล้วจะทำการเปลี่ยนที่อยู่ของ MAC (MAC address) ใน data frame ให้เป็นของเครื่องเซิร์ฟเวอร์ที่ได้ทำการเลือกภายในคลัสเตอร์ และจากนั้นจะทำการส่ง packet นั้นไปตามเครือข่าย LAN เมื่อเครื่องเซิร์ฟเวอร์ที่ได้เลือกได้รับ packet เครื่องเซิร์ฟเวอร์เครื่องนั้นก็จะพบว่า packet นั้นมีที่อยู่ปลายทางเป็นของอินเตอร์เฟสเสมือนของ loopback ของเครื่องมันเองและจากนั้นมันก็จะทำงานกับสัญญาณการร้องขอบริการ และจากนั้นก็จะทำการคืนผลลัพธ์ไปให้กับไคลเอนต์โดยตรง โดยที่เครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์แต่ละเครื่องนั้นจะมีอินเตอร์เฟสที่ทำการติดตั้งเป็น Virtual IP ที่จะไม่ทำการ ARP response มิฉะนั้นแล้วจะเกิดการชนกันของหมายเลขไอพีเพราะว่ามีหมายเลขไอพีซ้ำกันอยู่ภายในเครือข่ายเดียวกัน

วิธีการของ Direct Routing จะมีขั้นตอนการทำงานตามรูปที่ 3.7



รูปที่ 3.7 แสดงขั้นตอนการทำงานของ Linux Virtual Server via Direct Routing

จากรูปที่ 3.7 เครื่องโหลดบาลานซ์จะทำการเปลี่ยน MAC address ของ data frame ในสัญญาณการร้องขอบริการเป็นของเครื่องเซิร์ฟเวอร์ที่อยู่ภายในคลัสเตอร์และทำการส่ง packet นั้นไปให้กับไคลเอนต์โดยตรงผ่านทาง LAN ซึ่งเป็นเหตุผลที่ทำให้เครื่องโหลดบาลานซ์และเครื่องเซิร์ฟเวอร์ต่างๆภายในคลัสเตอร์จำเป็นที่ต้องอยู่ใน LAN วงเดียวกัน

3.2.4 ข้อเปรียบเทียบของวิธีการต่างๆ

ลักษณะเฉพาะของเทคนิคทางด้าน IP Load Balancing ที่ใช้ใน Linux Virtual Server ทั้ง 3 วิธีสรุปในตารางที่ 3.3

	NAT	IP Tunneling	Direct Routing
Server	Any	Tunneling	non-arp device
Server network	Private	LAN/WAN	LAN
Server number	Low (10~20)	High	High
Server gateway	Load balancer	Own router	Own router

ตารางที่ 3.3 แสดงข้อเปรียบเทียบของวิธีการต่างๆที่ใช้ใน Linux Virtual Server

Network Address Translation

วิธีการนี้เครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์สามารถที่จะใช้ระบบปฏิบัติการอะไรก็ได้ที่สนับสนุนโพรโตคอล TCP/IP และจะใช้ที่อยู่ไอพีเพียงหมายเลขเดียวในการติดต่ออินเตอร์เน็ตที่เครื่องโหลดบาลานซ์และที่เครื่องเซิร์ฟเวอร์แต่ละเครื่องภายในคลัสเตอร์จะใช้ที่อยู่ไอพีแบบส่วนตัว (Private IP Address)

ข้อเสียเปรียบคือความสามารถทางด้าน Scalability นั้นจะถูกจำกัด กล่าวคือเครื่องโหลดบาลานซ์อาจจะ bottleneck กับระบบทั้งหมด เมื่อจำนวนของเครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์มีจำนวนมากขึ้นถึงประมาณ 20 เครื่อง ซึ่งจะขึ้นอยู่กับ throughout ของเครื่องเซิร์ฟเวอร์แต่ละเครื่องภายในคลัสเตอร์ เหตุผลก็เนื่องมาจากว่า packet ของสัญญาณการร้องขอบริการ และ packet ของสัญญาณตอบกลับจำเป็นต้องทำการเขียนใหม่โดยเครื่องโหลดบาลานซ์ ซึ่งถ้าสมมุติว่าความยาวเฉลี่ยของ packet แบบ TCP เป็น 536 ไบต์ และเวลาเฉลี่ยที่ใช้ในการเขียน packet ใหม่ประมาณ 60 ไมโครวินาที แล้ว throughout ของเครื่องโหลดบาลานซ์จะเป็น 8.93 เมกกะไบต์ต่อวินาที ซึ่งถ้า throughout โดยเฉลี่ยของเครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์เป็น 600 กิโลไบต์ต่อวินาที จะทำให้เครื่องที่เป็นโหลดบาลานซ์สามารถทำงานได้กับเครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์ได้ทั้งหมด 15 เครื่อง

IP Tunneling

สำหรับการบริการทางอินเตอร์เน็ตโดยส่วนมากแล้ว packet ของสัญญาณร้องขอส่วนมากแล้วจะมีขนาดเล็ก และ packet ของสัญญาณตอบรับโดยปกติจะมีข้อมูลที่ต้องทำการส่งเป็นจำนวนมาก ซึ่งเครื่องโหลดบาลานซ์ของวิธีการนี้จะสามารถที่จะทำงานกับเซิร์ฟเวอร์ที่อยู่ภายในคลัสเตอร์ได้มากถึง 100 เครื่อง และระบบยังคงไม่ bottleneck ที่เครื่องโหลดบาลานซ์ สาเหตุก็เนื่องมาจากว่าเครื่องโหลดบาลานซ์จะทำหน้าที่เพียงรับสัญญาณร้องขอบริการและทำการส่งสัญญาณนี้ให้กับเครื่องเซิร์ฟเวอร์ที่อยู่ภายในคลัสเตอร์ และเครื่องเซิร์ฟเวอร์เครื่องนั้นจะทำการส่งผลลัพธ์หรือสัญญาณตอบกลับไปให้กับเครื่อง ไคลเอนต์ โดยตรง ดังนั้นวิธีการ IP Tunneling สามารถที่จะทำการสร้าง Virtual Server ที่ทำงานกับงานที่มีเป็นจำนวนมากได้ ซึ่งวิธีการนี้

เหมาะสมอย่างมากในการสร้าง Virtual Proxy Server ที่เครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์ทำหน้าที่รับสัญญาณร้องขอผ่านทางเครื่องโหลดบาลานซ์และทำการเข้าถึงอินเตอร์เน็ตได้โดยตรงเพื่อทำการอ่านข้อมูลและทำการคืนผลลัพธ์นั้นไปให้กับเครื่องไคลเอนต์โดยตรง

อย่างไรก็ตามวิธีการของ IP Tunneling จำเป็นต้องให้เครื่องที่อยู่ภายในคลัสเตอร์สนับสนุนกับโพรโตคอล IP Tunneling ซึ่งระบบปฏิบัติการลีนุกซ์ได้ให้การสนับสนุนกับโพรโตคอลนี้ เนื่องมาจากว่าโพรโตคอล IP Tunneling ได้มาเป็นมาตรฐานสำหรับทุกระบบปฏิบัติการ ดังนั้นวิธีการ IP Tunneling จึงน่าที่จะใช้ได้กับระบบปฏิบัติการอื่นๆที่ไม่ใช่ระบบปฏิบัติการลีนุกซ์ด้วย

Direct Routing

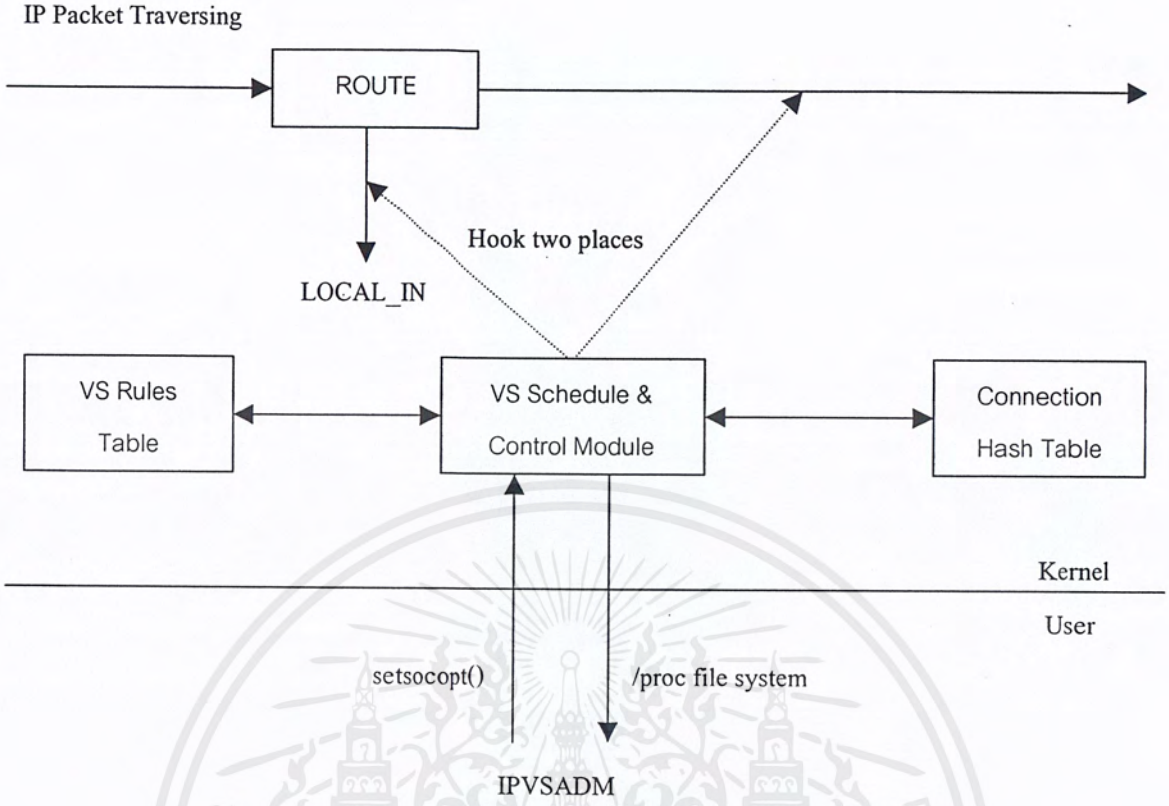
วิธีการนี้จะคล้ายกับวิธีการของ IP Tunneling โดยที่วิธีการนี้เครื่องโหลดบาลานซ์จะทำงานแค่เพียงครั้งเดียวของการติดต่อกับไคลเอนต์ ซึ่งส่วนของ packet ที่ตอบรับนั้นจะถูกส่งไปยังเครือข่ายอื่นที่ทำการเลือกเส้นทางให้ไปถึงเครื่อง ไคลเอนต์ ซึ่งจากเหตุผลข้างต้นทำให้สามารถเพิ่มความสามารถด้าน Scalability ของ Virtual Server ได้อย่างมาก

เมื่อเปรียบเทียบกับวิธีการของ IP Tunneling แล้ววิธีการ Direct Routing จะไม่มี overhead ที่เกิดจากวิธีการของ IP Tunneling แต่วิธีการของ Direct Routing จำเป็นต้องให้ระบบปฏิบัติการของทุกเครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์มีอินเตอร์เฟสเสมือนแบบ loopback ซึ่งจะไม่ต้องทำการ ARP response โดยที่เครื่องโหลดบาลานซ์กับทุกเครื่องที่เป็นเซิร์ฟเวอร์ภายในคลัสเตอร์จะต้องต่อกันเป็นเครือข่ายที่เป็น LAN วงเดียวกัน

3.2.5 วิธีการทำงานภายในระบบ

การสร้างระบบของ Linux Virtual Server จะเป็นไปตามรูปที่ 3.8 โดยที่ VS Schedule & Control Module จะเป็นโมดูลที่สำคัญของระบบ Linux Virtual Server โดยที่มันจะทำงานใน 2 ส่วนที่เป็นเส้นทางเดินของ IP packet ภายใน kernel เพื่อที่จะทำการตรวจจับและเขียน IP packet ใหม่เพื่อให้สนับสนุนกับวิธีการ Load Balancing ที่ใช้ โดยมันจะทำการดูที่ VS Rules ที่เป็นแบบ hash table สำหรับการติดต่อที่มีเข้ามาใหม่ภายในระบบ และทำการตรวจสอบกับ Connection Hash Table สำหรับการติดต่อกับไคลเอนต์ที่สร้างการติดต่อไว้แล้ว ส่วน IPVSADM จะเป็นโปรแกรมที่ผู้บริหารระบบใช้ในการจัดการปรับแต่งระบบของ Linux Virtual Server โดยที่มันจะใช้ฟังก์ชัน setsockopt เพื่อทำการเปลี่ยนแปลง Virtual Server Rules ที่อยู่ภายใน kernel และจะทำการอ่าน Virtual Server Rules ผ่านทางระบบไฟล์ /proc

จากรูปที่ 3.8 นั้น Connection Hash Table จะถูกออกแบบให้สามารถจัดการกับการติดต่อที่มีเข้ามาพร้อมกันเป็นจำนวนกว่าล้านการติดต่อ และส่วนที่บันทึกในตารางแต่ละส่วนของการติดต่อจะใช้เนื้อที่เพียง 128 ไบต์ในเครื่องโหลดบาลานซ์ ยกตัวอย่างเช่น เครื่องโหลดบาลานซ์ที่มีหน่วยความจำ 256 เมกกะไบต์สามารถที่จะใช้กับการติดต่อที่มีจำนวนประมาณ 2 ล้านการติดต่อได้อย่างพร้อมกัน ขนาดของ hash table สามารถที่จะเปลี่ยนแปลงได้โดยผู้ใช้งานเพื่อให้เหมาะสมกับงานแต่ละประเภทที่นำไปใช้งาน และส่วนของไคลเอนต์ ที่ใช้เป็น hash key นั้นจะเป็น <protocol, address, port> ซึ่งจะส่งผลให้การชนกันของ hash key มีน้อย ทำให้การทำงานของระบบดีขึ้น



รูปที่ 3.8 แสดงการทำงานภายในของระบบ Linux Virtual Server

3.3 Connection Scheduling

ในระบบ Linux Virtual Server จะมีวิธีการกระจายการร้องขอบริการไปให้กับเครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์หรือที่เรียกว่า Scheduling Algorithm อยู่ทั้งหมด 4 วิธีคือ

1. Round – Robin Scheduling
2. Weighed Round – Robin Scheduling
3. Least – Connection Scheduling
4. Weighted Least – Connection Scheduling

จาก Scheduling Algorithm ข้างบนนั้นสองวิธีแรกจะไม่ได้คำนึงถึงข้อมูลภาระงานที่ทำงานอยู่ในแต่ละเครื่องของเซิร์ฟเวอร์ที่อยู่ภายในคลัสเตอร์ ส่วนสองวิธีหลังนั้นจะทำการนับจำนวนการติดต่อที่กำลังทำงานอยู่ภายในแต่ละเครื่องเซิร์ฟเวอร์ และจะทำการกระจายงานไปให้เครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์โดยคำนึงถึงจำนวนการติดต่อเป็นเกณฑ์ในการกระจายงาน

3.3.1 Round – Robin Scheduling

วิธีการทำงานของ Round - Robin Scheduling คือเมื่อเครื่องที่ทำหน้าที่เป็นโหลดบาลานซ์ได้รับการร้องขอบริการเข้ามาก็จะทำการกระจายการร้องขอบริการนั้นให้กับเซิร์ฟเวอร์ที่อยู่ภายในคลัสเตอร์แบบวนไปเรื่อยๆ โดยที่จะไม่ได้พิจารณาถึงจำนวนการติดต่อหรือเวลาตอบสนองการร้องขอบริการของเซิร์ฟเวอร์แต่ละเครื่องที่อยู่ภายในคลัสเตอร์

การทำงานของวิธีการนี้จะคล้ายกับหลักการของ Round – Robin DNS ซึ่งหลักการของ Round – Robin DNS นั้นจะทำการจับคู่ชื่อโดเมนหนึ่งกับหมายเลขไอพีที่มากกว่าหนึ่ง ซึ่งหน่วยย่อยของการ Scheduling แบบนี้จะอยู่ในรูปของเครื่อง และการ cache ของ Round – Robin DNS นั้นจะทำให้ Round – Robin Scheduling ที่ใช้ไม่ได้ผลตามที่ต้องการ ซึ่งจะส่งผลให้เกิดการกระจายงานที่ไม่สมดุลในระหว่างเครื่องเซิร์ฟเวอร์ที่อยู่ภายในคลัสเตอร์ ซึ่งเมื่อเทียบกับ Round – Robin Scheduling ของ Linux Virtual Server แล้ว หน่วยย่อยของการ Scheduling จะอยู่ในรูปของการติดต่อ ซึ่งมันจะมีประสิทธิภาพมากกว่าวิธีการของ Round – Robin DNS เนื่องจากหน่วยย่อยในการทำงานมีความละเอียดมากกว่า

3.3.2 Weighted Round – Robin Scheduling

วิธีการทำงานของ Weighted Round - Robin Scheduling คือจะทำงานคล้ายกับ Round – Robin Scheduling แต่เราสามารถที่จะทำการกระจายงานโดยคำนึงถึงประสิทธิภาพในการทำงานที่แตกต่างกันของเซิร์ฟเวอร์ต่างๆที่อยู่ภายในคลัสเตอร์ โดยวิธีการนี้จะทำการกำหนดค่าตัวเลขที่บ่งบอกถึงประสิทธิภาพในการทำงานของเซิร์ฟเวอร์แต่ละเครื่องที่อยู่ภายในคลัสเตอร์

ขั้นตอนการทำงานของ Weighted Round – Robin Scheduling จะเป็นไปดังนี้คือ ถ้าสมมุติว่ามีเครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์จำนวน n เครื่องคือ $S = \{S_0, S_1, ..., S_{n-1}\}$ โดยที่ดัชนี i จะแทนเครื่องเซิร์ฟเวอร์ที่เพิ่งได้รับการเลือกในการกระจายงานโดยเครื่องโหลดบาลานซ์ และตัวแปร cw จะแทนค่าถ่วงน้ำหนักปัจจุบันในขณะนั้น โดยที่ตัวแปร i และ cw จะมีค่าเริ่มต้นเป็นศูนย์ ถ้าทุก $W(S_i)$ มีค่าเท่ากับศูนย์ แสดงว่าไม่มีเครื่องเซิร์ฟเวอร์ภายในคลัสเตอร์ทำงานได้ เพราะฉะนั้นทุกการติดต่อที่เข้ามายัง Virtual Server จะไม่ทำงาน ซึ่ง algorithm จะเป็นไปตามข้างล่างนี้

```
while (1) {
    if ( i == 0 ) {
        cw = cw – 1;
        if (cw <= 0 ) {
            set cw the maximum weight of S;
            if ( cw == 0 ) return NULL;
        }
    } else i = ( i + 1 ) mod n ;
    if ( W(Si) >= cw ) return Si ;
}
```

วิธีการ scheduling แบบนี้ทุกเซิร์ฟเวอร์ที่มีค่าถ่วงน้ำหนักมากจะทำการรับสัญญาณร้องขอก่อนและมีการติดต่อที่มากกว่าเครื่องเซิร์ฟเวอร์ที่มีค่าถ่วงน้ำหนักน้อยกว่า และเครื่องเซิร์ฟเวอรที่มีค่าถ่วงน้ำหนักเท่ากันก็จะได้รับการติดต่อของสัญญาณการร้องที่มีจำนวนเท่ากันด้วย เช่น ถ้ามีเซิร์ฟเวอร์ A, B, และ C ซึ่งมีค่าถ่วงน้ำหนักเป็น 4, 3 และ 2 ตามลำดับ แล้วการ Scheduling ที่ดีในช่วงเวลาหนึ่งจะเป็นดังนี้คือ ABCABCABA

วิธีการแบบ Weighted Round – Robin Scheduling นั้นจะไม่ได้คำนึงถึงจำนวนการติดต่อของแต่ละเซิร์ฟเวอร์ที่อยู่ภายในคลัสเตอร์ ซึ่งทำให้มี overhead น้อย และส่งผลให้ระบบคลัสเตอร์สามารถมีจำนวนเซิร์ฟเวอร์ภายในคลัสเตอร์ได้มาก แต่อย่างไรก็ตามวิธีการนี้อาจนำไปสู่การกระจายงานอย่างไม่สมดุลภายในระบบคลัสเตอร์ได้ถ้าภาระของการร้องขอบริการมีความแตกต่างกันมาก กล่าวคือภาระการร้องขอบริการจำนวนมากที่มีการทำงานมากอาจจะถูกส่งไปให้กับเซิร์ฟเวอร์เครื่องหนึ่งที่อยู่ภายในคลัสเตอร์มากกว่าเครื่องอื่นได้

3.3.3 Least - Connection Scheduling

วิธีการทำงานของ Least – Connection Scheduling คือจะทำการกระจายงานไปให้กับเซิร์ฟเวอร์ที่มีจำนวนการติดต่ออยู่กับการร้องขอบริการที่มีอยู่น้อยที่สุด โดยที่วิธีการนี้จะต้องทำการนับการติดต่อที่ยังทำงานอยู่ภายในเซิร์ฟเวอร์แต่ละเครื่องที่อยู่ภายในคลัสเตอร์ ซึ่งวิธีการแบบนี้จะเหมาะสำหรับระบบคลัสเตอร์ที่เซิร์ฟเวอร์แต่ละเครื่องมีความสามารถในการทำงานที่ใกล้เคียงกัน และเหมาะสำหรับภาระการร้องขอบริการที่มีการทำงานที่แตกต่างกันมาก เพราะว่าภาระการร้องขอบริการที่มีการทำงานมากจะไม่มีโอกาสได้ส่งไปยังเซิร์ฟเวอร์เครื่องหนึ่งเครื่องใดที่อยู่ภายในคลัสเตอร์เพียงเครื่องเดียว

การกระจายงานอย่างนี้จะทำงานได้ไม่ดีเมื่อแต่ละเซิร์ฟเวอร์ที่อยู่ภายในคลัสเตอร์มีประสิทธิภาพในการทำงานที่แตกต่างกัน สาเหตุก็เนื่องมาจากสถานะ TIME_WAIT ของโพรโตคอล TCP ซึ่งโดยทั่วไปจะมีค่าสองนาที ซึ่งในเวลาสองนาทีนี้เว็บไซต์ที่เป็นที่นิยมสามารถรับการร้องขอบริการที่มีเข้ามาได้มากกว่าจำนวนหนึ่งพันการร้องขอบริการ ตัวอย่างเช่น เมื่อเซิร์ฟเวอร์เครื่องหนึ่งมีประสิทธิภาพในการทำงานเป็นสองเท่าของอีกเครื่องหนึ่งที่อยู่ภายในคลัสเตอร์เดียวกัน เครื่องที่ทำงานเร็วกว่ากำลังทำงานกับการร้องขอบริการที่มีจำนวนเป็นพันการร้องขอบริการ และได้เก็บงานการร้องขอบริการนั้นไว้ในสถานะ TIME_WAIT ของโพรโตคอล TCP ส่วนเครื่องที่ทำงานช้ากว่าได้ทำงานกับการร้องขอบริการที่มีมาเรื่อยๆ และในที่สุดก็ได้ทำงานกับการร้องขอบริการที่มีเข้ามาเป็นจำนวนหนึ่งพันการร้องขอบริการเสร็จ และยังคงรับสัญญาณร้องขอที่มีเข้ามาได้เรื่อยๆ ด้วยเหตุนี้การทำงานของ Least – Connection Scheduling จะทำการกระจายงานไปให้เซิร์ฟเวอร์ต่างๆ ภายในคลัสเตอร์ได้อย่างไม่สมดุลเมื่อเซิร์ฟเวอร์แต่ละเครื่องภายในคลัสเตอร์มีประสิทธิภาพในการทำงานที่แตกต่างกัน

3.3.4 Weighted Least – Connection Scheduling

วิธีการทำงานของ Weighted Least – Connection Scheduling คือจะทำงานคล้ายกับ Least – Connection Scheduling แต่เราสามารถที่จะทำการกระจายงานโดยคำนึงถึงประสิทธิภาพในการทำงานที่แตกต่างกันของเซิร์ฟเวอร์ต่างๆที่อยู่ภายในคลัสเตอร์ โดยวิธีการนี้จะทำการกำหนดค่าตัวเลขที่บ่งบอกถึงประสิทธิภาพในการทำงานของเซิร์ฟเวอร์แต่ละเครื่องที่อยู่ภายในคลัสเตอร์ ซึ่งถ้าจำนวนการติดต่อของการร้องขอบริการเท่ากันเซิร์ฟเวอร์ที่มีค่าถ่วงน้ำหนักสูงกว่าเมื่อเทียบกับเครื่องอื่นจะทำการรับภาระการร้องขอบริการมากเมื่อเทียบกับเครื่องอื่นด้วย ซึ่งผู้ดูแลระบบสามารถทำการกำหนดค่าถ่วงน้ำหนักนี้ให้กับแต่ละเซิร์ฟเวอร์ที่อยู่ภายในคลัสเตอร์ ซึ่งการกระจายงานจะทำโดยคำนึงถึงจำนวนการติดต่อของเซิร์ฟเวอร์แต่ละเครื่องและค่าถ่วงน้ำหนักที่กำหนดให้กับแต่ละเซิร์ฟเวอร์

การทำงานของ Weighted Least – Connection Scheduling จะเป็นไปดังนี้ สมมุติว่ามีเซิร์ฟเวอร์จำนวน n เครื่อง แต่ละเซิร์ฟเวอร์ i มีค่าถ่วงน้ำหนักเป็น W_i (i = 1, … ,n) และจำนวนการติดต่อของแต่ละเซิร์ฟเวอร์ i เป็น C_i (i = 1, … ,n) โดยที่ผลรวมของการติดต่อทั้งหมดของ C_i (i = 1, … ,n) เป็น T แล้วการติดต่อครั้งต่อไปจะถูกส่งไปให้กับเซิร์ฟเวอร์ j ซึ่ง

$$(C_j / T) / W_j \quad = \quad \text{MIN} \{(C_i / T) / W_i\} \ (i = 1, \ldots, n)$$

เพราะว่า T เป็นค่าคงที่ จึงเขียนให้อยู่ในรูปที่ง่ายได้เป็น

$$C_j / W_j \quad = \quad \text{MIN} \{C_i / W_i\} \ (i = 1, \ldots, n)$$

เนื่องมาจากว่าใน kernel ของลีนุกซ์จะไม่มีจำนวนแบบ float การเปรียบเทียบค่าของ $C_j/W_j > C_i/W_i$ จะเปลี่ยนเป็น $C_j * W_i > C_i * W_j$ เพราะว่าค่าถ่วงน้ำหนักจะมีค่าเริ่มต้นเป็นหนึ่งและจะไม่มีทางมีค่าเป็นศูนย์

บทที่ 4
การติดตั้ง

4.1 Apache Web Server

การจะได้มาซึ่งตัวโปรแกรม Apache นั้น มีอยู่ด้วยกัน 2 ทางเลือก คือ ใช้ไฟล์ rpm ซึ่งมีอยู่ภายในตัวลีนุกซ์ หรือการนำ source code มาทำการ compile เอง ซึ่งสามารถ download ได้ที่ http://www.apache.org และเมื่อได้ไฟล์มาแล้วให้ทำดังนี้

แบบที่เป็น rpm

วิธีนี้เป็นวิธีที่สะดวก โดยที่เราไม่ต้องลำบากมาทำการ compile เอง

- ติดตั้งโดยใช้คำสั่ง rpm ได้เลย

แบบที่เป็น Source Code

วิธีนี้ใช้ในกรณีที่ต้องการเพิ่ม module ให้กับตัวโปรแกรมเพื่อเพิ่มประสิทธิภาพบางอย่าง เช่น เพิ่ม mod_auth_digest ทำให้สามารถใช้ Digest Authentication ได้ ซึ่งมีความปลอดภัยมากกว่า User Authentication แบบเดิม

การจะ compile ได้นั้นจะต้องมีทั้ง gcc และ perl 5 เสียก่อน แล้วทำตามขั้นตอนดังนี้

- cd path/to/file
- gzip –cd filename | tar xvf –
- cd path/to/Apache-1.3.12
- แก้ไขไฟล์ Configuration ในไดเรกทอรี src โดย uncomment module ที่ต้องการเพิ่ม
- cd path/to/Apache-1.3.12
- ./configure - -prefix = path/to/install/Apache
- make install
- โปรแกรมได้ติดตั้งลงใน path/to/install/Apache แล้ว (ต่อไปจะใช้ path/to/Apache แทน)

Directive ที่สำคัญในไฟล์ httpd.conf

ไฟล์ httpd.conf จะอยู่ใน path/to/Apache/etc ซึ่งในไดเรกทอรีนี้มีไฟล์ที่สำคัญ 3 ไฟล์ คือ httpd.conf , srm.conf , access.conf ซึ่งไดเรกทีฟทั้งหมดจะแยกไว้ทั้ง 3 ไฟล์ หรืออยู่ใน httpd.conf ไฟล์เดียวก็ได้ ในที่นี้เก็บไว้ที่ httpd.conf ไฟล์เดียว โดยใส่ comment ที่ 2 ไดเรกทีฟนี้

- # ResourceConfig conf/srm.conf
- # AccessConfig conf/access.conf

ServerType

ไดเรกทีฟ ServerType เป็นการบอกชนิดของ Server ซึ่งมีอยู่ 2 ชนิด คือ

- standalone
- inetd

เช่น ServerType standalone ซึ่งชนิด standalone จะเปิด server ไว้ตลอดเวลา ส่วน inetd จะทำการเริ่มการทำงานของเซิร์ฟเวอร์เมื่อมีการเรียกจากไคลเอนต์เท่านั้น

Timeout

ไดเรกทีฟ Timeout คือ เวลาเป็นวินาทีก่อนที่เครื่องเซิร์ฟเวอร์จะทำาการส่งสัญญาณว่าเครื่องเซิร์ฟเวอร์ไม่ว่าง เช่น Timeout 300

Maxclient

ไดเรกทีฟ Maxclient คือจำนวนไคลเอนต์สูงสุดที่สามารถใช้งานได้พร้อมกัน เช่นMaxclient 150

Listen

ไดเรกทีฟ Listen เป็นการบอกหมายเลขไอพีหรือและหมายเลข port หรือเฉพาะ port ที่ Apache จะไปคอยฟังสัญญาณ เช่น Listen 80 หรือ Listen 161.246.11.49:8080

Port

ไดเรกทีฟ Port เป็นการบอก port ที่ Apache จะทำการเชื่อมต่อด้วย อย่าสับสนกับไดเรกทีฟ Listen เช่น Port 80

User

ไดเรกทีฟ User จะบอกถึงผู้ใช้ที่สามารถเข้าไปดูในเว็บได้ เช่น User nobody

Group

ไดเรกทีฟ Group บอกถึงกลุ่มของผู้ใช้ที่สามารถเข้าไปดูในเว็บได้ เช่น Group nobody

DocumentRoot

ไดเรกทีฟ DocumentRoot จะบอกถึงไดเรกทอรีที่เก็บโฮมเพจหน้าแรกของเว็บไซต์นั้น เช่น DocumentRoot "path/to/Apache/lib/htdocs"

ScriptAlias

ไดเรกทีฟ ScriptAlias บอกไดเรกทอรีที่เก็บ cgi script เช่น

ScriptAlias /cgi-bin/ "path/to/Apache/lib/cgi-bin/"

Redirect

ไดเรกทีฟ Redirect บอก URL ที่ต้องการย้ายไป เช่น Redirect /yahoo http://www.yahoo.com

AddType

ไดเรกทีฟ AddType และ AddHandler ใช้เมื่อต้องการจะใช้ไฟล์ชนิดอื่นนอกเหนือจากไฟล์ html เช่น AddType text/html .shtml

AddHandler

ไดเรกทีฟ AddHandler จะใช้บอกตัวจัดการกับไฟล์ที่ใช้นอกเหนือจากไฟล์ html เช่น

AddHandler server-parsed .shtml

เทคนิคการใช้งาน Apache Web Server

สมมติให้ Web Server มี IP Address เป็น 161.246.11.49 และมีชื่อ banana.com ซึ่งจะมีขั้นตอนดังนี้

1) การใช้งาน User home directory

- ดูในไดเรกทีฟ UserDir ว่ามีค่าอะไร (ปกติจะเป็น public_html)

```
<IfModule mod_userdir.c>
    UserDir public_html
</IfModule>
```

- สมมติมียูสเซอร์ teru ให้สร้างไดเรกทอรี public_html ไว้ใน /home/teru

```
cd /home/teru
mkdir public_html
```

- สร้างไฟล์ index.html ไว้ใน public_html
- ดูโฮมเพจของผู้ใช้งาน teru ได้จาก http://banana.com/~teru

2) การใช้งาน cgi

- ดูไดเรกทีฟ ScriptAlias ว่าอยู่ที่ไดเรกทอรีใด (ปกติคือ path/to/Apache/lib/cgi-bin)

ScriptAlias /cgi-bin/ "path/to/Apache/lib/cgi-bin"

- นำโปรแกรม cgi ที่จะใช้ไปไว้ที่ path/to/Apache/lib/cgi-bin (สมมติชื่อ teru.cgi)
- เปลี่ยนไฟล์ teru.cgi ให้อยู่ในโหมดที่ execute ได้

chmod 755 teru.cgi

- เรียกใช้ cgi ได้จาก http://banana.com/cgi-bin/teru.cgi

3) การทำ Redirection

Redirection คือ การย้าย URL จากที่หนึ่งไปยังอีกที่หนึ่ง เช่น เราต้องการนำเว็บไซต์ของเราซึ่งอยู่ที่ http://161.246.11.49 ไปฝากไว้ที่ http://come.to โดยใช้ชื่อ banana สิ่งที่ต้องทำ คือ

- ที่ Web Server ของ come.to (สมมติว่าคือ Apache) เมื่อเราติดต่อขอฝากแล้ว เขาจะเพิ่มข้อมูลที่ไดเรกทีฟ Redirection ดังนี้

Redirection /banana http://161.246.11.49

- สามารถเข้าไปเว็บไซต์ของเราได้ที่ http://come.to/banana

4) การใช้งาน SSI (Server Side Include)

SSI คือ สคริปต์ที่แปะติดอยู่กับ code html และประมวลผลที่ server ซึ่งจะทำให้เว็บไซต์ที่ได้เป็น dynamic html

- เอา comment ออกที่ 2 ไดเรกทีฟ ดังนี้

AddType text/html .shtml

AddHandler server-parsed .shtml

- สมมติเราใช้ SSI ที่ DocumentRoot ให้เพิ่ม Includes ที่ไดเรกทีฟ Options ส่วนอื่นคงเดิม

<Directory "/path/to/Apache/lib/htdocs">

Options Includes

- นำไฟล์ SSI ของเรา สมมติชื่อ ssi.shtml ไปไว้ที่ path/to/Apache/lib/htdocs
- เรียกใช้งาน SSI ได้ที่ http://161.246.11.49/ssi.shtml

5) การทำ Virtual Host

Virtual Host คือ การทำให้ Web Server ตัวเดียวมีได้หลายเว็บไซต์ เช่น

ก. เครื่องเดียวกันแต่มีหลาย IP Address ใช้ IP ที่ต่างกันเป็น Virtual Host

ข. IP เดียวกันแต่มีหลายชื่อ ซึ่งชื่อนั้นๆจะต้องมีอยู่จริงๆ เช่น ให้ 161.246.11.49 มีชื่อเป็น banana.crsc.kmitl.ac.th และ teru.com ใช้แต่ละชื่อเป็น Virtual Host

ค. IP เดียวกันแต่เรียกใช้โพรโตคอล http ที่ port แตกต่างกัน เราจะใช้กรณีนี้เป็นตัวอย่าง ภายในไดเรกทีฟ VirtualHost จะประกอบไปด้วยไดเรกทีฟอื่นๆที่อยู่ภายนอกได้ เช่น ServerAdmin, Port, DocumentRoot, ServerName จะเห็นได้ว่าถ้ากำหนด DocumentRoot ต่างกัน หน้าโฮมเพจที่ปรากฏออกมาก็จะต่างกันด้วย

- กำหนดให้ server คอยฟังสัญญาณที่ 2 port คือ 80 และ 8080

Listen 80

Listen 8080

- ให้เว็บไซต์หลักใช้ port ที่หมายเลข 80

Port 80

- ภายใน VirtualHost กำหนดให้ใช้ port 8080 และเปลี่ยน DocumentRoot ไปที่อื่น

<VirtualHost 161.246.11.49:8080>

Port 8080

DocumentRoot "/path/to/Apache/lib/virtual"

</VirtualHost>

- สร้างไดเรกทอรี virtual ไว้ที่ path/to/Apache/lib

cd path/to/Apache/lib

mkdir virtual

- สร้างไฟล์ index.html ไว้ในไดเรกทอรี virtual
- เข้าดู Virtual Host ได้ทาง http://161.246.11.49:8080

การ Start และ Stop ตัว Apache Web Server

ก. /path/to/Apache/bin/apachectl start

ข. /path/to/Apache/bin/apachectl stop

ค. /path/to/Apache/bin/apachectl restart

ข้อควรจำ ทุกครั้งที่มีการแก้ไขไฟล์ httpd.conf จะต้องทำการ restart Web Server ใหม่

4.2 การติดตั้งระบบ Linux Virtual Server โดยการ patch kernel

วิธีการสร้าง kernel เพื่อให้ Linux Virtual Server ทำงานได้

ตอนแรกให้ทำการ patch kernel ซึ่งโปรแกรมที่ทำการ patch สามารถทำการ download ได้จาก

http://www.linuxvirtualserver.org/software/index.html

จากนั้นทำการ compile kernel โดยที่ทางเลือกเหล่านี้ต้องได้รับการเลือกไว้

Kernel Compile Options :

Code maturity level options --->

[*] Prompt for development and/or incomplete code/drivers

Networking options --->

[*] Network firewalls

...

[*] IP: forwarding/gatewaying

...

[*] IP: firewalling

...

[*] IP: masquerading

...

[*] IP: masquerading virtual server support (EXPERIMENTAL)

(12) IP masquerading table size (the Nth power of 2)

<M> IPVS: round-robin scheduling(NEW)

<M> IPVS: weighted round-robin scheduling(NEW)

<M> IPVS: weighted least-connection scheduling(NEW)

<M> IPVS: persistent client connection scheduling(NEW)

จากนั้นทำการสร้าง kernel ใหม่ โดยที่จะต้องทำการสร้างอย่างถูกวิธี จากนั้นทำการ update ระบบของเรา และก็ reboot เครื่องใหม่

วิธีการสร้างระบบ Linux Virtual Server via NAT

หลังจากที่ทำการ patch kernel แล้วเราจะสามารถใช้คำสั่ง ipvsadm ซึ่งจะใช้สำหรับทำการติดตั้งระบบ โดยจะใช้กำหนดกฎใน virtual server ตัวอย่างเช่น เราต้องการติดตั้งระบบแบบ NAT ที่มี Virtual IP Address เป็น 161.246.5.99 มีเครื่องโหลดบาลานซ์ที่มีหมายเลขไอพีเป็น 161.246.5.100 ซึ่งทำการติดต่อทาง port หมาย

เลข 80 โดยใช้ schedule เป็น Weighted-Least Connection มีเครื่องเซิร์ฟเวอร์ในคลัสเตอร์ที่มีหมายเลขไอพีเป็น 192.168.0.2 ทำการติดต่อทาง port หมายเลข 80 มีค่าถ่วงน้ำหนักเป็น 1 และมีเครื่องเซิร์ฟเวอร์ในคลัสเตอร์ที่มีหมายเลขไอพีเป็น 192.168.0.3 ที่ทำการติดต่อทาง port หมายเลข 8000 ที่มีค่าถ่วงน้ำหนักเป็น 2 ซึ่งคำสั่งที่ใช้จะเป็น

ที่เครื่องโหลดบาลานซ์จะใช้คำสั่งต่อไปนี้

```
ifconfig eth0 161.246.5.100 netmask 255.255.255.0 broadcast 161.246.5.255 up
route add -net 161.246.5.0 netmask 255.255.255.0 dev eth0
ifconfig eth1 192.168.0.1 netmask 255.255.255.0 broadcast 192.168.0.255 up
route add -net 192.168.0.0 netmask 255.255.255.0 dev eth1
route add –host 192.168.0.1 dev eth1
ifconfig eth0:0 161.246.5.99 netmask 255.255.255.255 broadcast 161.246.5.99 up
route add -host 161.246.5.99 dev eth0:0
echo 1 > /proc/sys/net/ipv4/ip_forward
ipvsadm -A -t 161.246.5.99:80 -s wlc
ipvsadm -a  -t 161.246.5.99:80 -r 192.168.0.2:80 –m –w 1
ipvsadm -a  -t 161.246.5.99:80 -r 192.168.0..3:8000 –m –w 2
```

ที่เครื่องเซิร์ฟเวอร์ที่มีหมายเลขไอพีเป็น 192.168.0.2 จะใช้คำสั่งต่อไปนี้

```
ifconfig eth0 192.168.0.2 netmask 255.255.255.0 broadcast 192.168.0.255 up
route add -net 192.168.0.0 netmask 255.255.255.0 dev eth0
route add -host 192.168.0.2 dev eth0
route add default gw 192.168.0.1 eth0
```

ที่เครื่องเซิร์ฟเวอร์ที่มีหมายเลขไอพีเป็น 192.168.0.3 จะใช้คำสั่งต่อไปนี้

```
ifconfig eth0 192.168.0.3 netmask 255.255.255.0 broadcast 192.168.0.255 up
route add -net 192.168.0.0 netmask 255.255.255.0 dev eth0
route add -host 192.168.0.3 dev eth0
route add default gw 192.168.0.1 eth0
```

วิธีการสร้างระบบ Linux Virtual Server via IP Tunneling

การติดตั้งจะเป็นดังนี้ ตัวอย่างเช่น เราต้องการติดตั้งระบบแบบ IP Tunneling ที่มี Virtual IP Address เป็น 161.246.5.99 มีเครื่องโหลดบาลานซ์ที่มีหมายเลขไอพีเป็น 161.246.5.100 ซึ่งทำการติดต่อทาง port หมายเลข 80 โดยใช้ schedule เป็น Weighted-Least Connection มีเครื่องเซิร์ฟเวอร์ในคลัสเตอร์ที่มีหมายเลขไอพีเป็น

161.246.5.101 ทำการติดต่อทาง port หมายเลข 80 มีค่าถ่วงน้ำหนักเป็น 1 และมีเครื่องเซิร์ฟเวอร์ในคลัสเตอร์ที่มีหมายเลขไอพีเป็น 161.246.5.102 ที่ทำการติดต่อทาง port หมายเลข 8000 ที่มีค่าถ่วงน้ำหนักเป็น 2 ซึ่งคำสั่งที่ใช้จะเป็นดังนี้

ที่เครื่องโหลดบาลานซ์จะใช้คำสั่งต่อไปนี้

```
ifconfig eth0 161.246.5.100 netmask 255.255.255.0 broadcast 161.246.5.255 up
route add -net 161.246.5.0 netmask 255.255.255.0 dev eth0
ifconfig eth0:0 161.246.5.99 netmask 255.255.255.255 broadcast 161.246.5.99 up
route add -host 161.246.5.99 dev eth0:0
echo 1 > /proc/sys/net/ipv4/ip_forward
ipvsadm -A -t 161.246.5.99:80 -s wlc
ipvsadm -a  -t 161.246.5.99:80 -r 161.246.5.101:80 –i –w 1
ipvsadm -a  -t 161.246.5.99:80 -r 161.246.5.102:8000 –i –w 2
```

ที่เครื่องเซิร์ฟเวอร์ที่มีหมายเลขไอพีเป็น 161.246.5.101 จะใช้คำสั่งต่อไปนี้

```
ifconfig eth0 161.246.5.101 netmask 255.255.255.0 broadcast 161.246.5.255 up
route add -net 161.246.5.0 netmask 255.255.255.0 dev eth0
ifconfig tunl0 161.246.5.99 netmask 255.255.255.255 broadcast 161.246.5.99 up
route add -host 161.246.5.99 dev tunl0
```

ที่เครื่องเซิร์ฟเวอร์ที่มีหมายเลขไอพีเป็น 161.246.5.102 จะใช้คำสั่งต่อไปนี้

```
ifconfig eth0 161.246.5.102 netmask 255.255.255.0 broadcast 161.246.5.255 up
route add -net 161.246.5.0 netmask 255.255.255.0 dev eth0
ifconfig tunl0 161.246.5.99 netmask 255.255.255.255 broadcast 161.246.5.99 up
route add -host 161.246.5.99 dev tunl0
```

วิธีการสร้างระบบ Linux Virtual Server via Direct Routing

การติดตั้งจะเป็นดังนี้ ตัวอย่างเช่น เราต้องการติดตั้งระบบแบบ Direct Routing ที่มี Virtual IP Address เป็น 161.246.5.99 มีเครื่องโหลดบาลานซ์ที่มีหมายเลขไอพีเป็น 161.246.5.100 ซึ่งทำการติดต่อทาง port หมายเลข 80 โดยใช้ schedule เป็น Weighted-Least Connection มีเครื่องเซิร์ฟเวอร์ในคลัสเตอร์ที่มีหมายเลขไอพีเป็น 161.246.5.101 ทำการติดต่อทาง port หมายเลข 80 มีค่าถ่วงน้ำหนักเป็น 1 และมีเครื่องเซิร์ฟเวอร์ในคลัสเตอร์ที่มีหมายเลขไอพีเป็น 161.246.5.102 ที่ทำการติดต่อทาง port หมายเลข 8000 ที่มีค่าถ่วงน้ำหนักเป็น 2 ซึ่งคำสั่งที่ใช้จะเป็นดังนี้

ที่เครื่องโหลดบาลานซ์จะใช้คำสั่งต่อไปนี้

```
ifconfig eth0 161.246.5.100 netmask 255.255.255.0 broadcast 161.246.5.255 up
route add -net 161.246.5.0 netmask 255.255.255.0 dev eth0
ifconfig eth0:0 161.246.5.99 netmask 255.255.255.255 broadcast 161.246.5.99 up
route add -host 161.246.5.99 dev eth0:0
echo 1 > /proc/sys/net/ipv4/ip_forward
ipvsadm -A -t 161.246.5.99:80 -s wlc
ipvsadm -a  -t 161.246.5.99:80 -r 161.246.5.101:80 –g –w 1
ipvsadm -a  -t 161.246.5.99:80 -r 161.246.5.102:8000 –g –w 2
```

ที่เครื่องเซิร์ฟเวอร์ที่มีหมายเลขไอพีเป็น 161.246.5.101 จะใช้คำสั่งต่อไปนี้

```
ifconfig eth0 161.246.5.101 netmask 255.255.255.0 broadcast 161.246.5.255 up
route add -net 161.246.5.0 netmask 255.255.255.0 dev eth0
ifconfig lo:0 161.246.5.99 netmask 255.255.255.255 broadcast 161.246.5.99 up
route add -host 161.246.5.99 dev lo:0
```

ที่เครื่องเซิร์ฟเวอร์ที่มีหมายเลขไอพีเป็น 161.246.5.102 จะใช้คำสั่งต่อไปนี้

```
ifconfig eth0 161.246.5.102 netmask 255.255.255.0 broadcast 161.246.5.255 up
route add -net 161.246.5.0 netmask 255.255.255.0 dev eth0
ifconfig lo:0 161.246.5.99 netmask 255.255.255.255 broadcast 161.246.5.99 up
route add -host 161.246.5.99 dev lo:0
```

4.3 การติดตั้งระบบ Linux Virtual Server โดยใช้โปรแกรม Piranha Web Interface

Piranha Web Interface เป็นโปรแกรมที่มีประโยชน์มากในการปรับปรุงเปลี่ยนแปลงระบบคลัสเตอร์ที่เราจะใช้งานและยังเป็นโปรแกรมที่ใช้ดูสถานะการทำงานของเครื่องเซิร์ฟเวอร์ต่างๆภายในคลัสเตอร์ โดยที่นำเสนอวิธีติดตั้งเป็นแบบเชิงโครงสร้างในการสร้างไฟล์ configuration ที่จำเป็นต้องใช้ภายในระบบคลัสเตอร์แบบ Linux Virtual Server โดยการใช้งานของโปรแกรมนี้จะติดต่อกับผู้ใช้งานในรูปของเว็บเพจซึ่งเข้าใจได้ง่ายไม่ซับซ้อนประกอบกับเป็นที่คุ้นเคยของคนทั่วไป รวมทั้งการลงโปรแกรมก็ทำได้ง่ายมาก เพราะว่าโปรแกรมนี้จะอยู่ในรูปของ RPM package

ข้อได้เปรียบของการติดตั้งระบบ Linux Virtual Server โดยการใช้ Piranha Web Interface นั้นคือจะช่วยให้การติดตั้งง่ายขึ้น เราไม่จำเป็นต้องรู้รูปแบบที่ใช้ในการเขียนไฟล์ configuration เพราะว่าโปรแกรม Piranha Web Interface จะเป็นผู้ทำให้หมด ซึ่งไฟล์ configuration เหล่านี้จะมีรูปแบบในการเขียน ซึ่งการเขียน

ไฟล์เหล่านี้ต้องตรงตามรูปแบบที่มันกำหนด ถ้าไม่เป็นไปตามรูปแบบแล้วระบบจะไม่เข้าใจก็จะทำให้ระบบของคลัสเตอร์ทำงานผิดความต้องการหรืออาจจะไม่สามารถทำงานได้เลยก็เป็นได้

Piranha Web Interface Requirements

โปรแกรม Piranha Web Interface จะทำงานได้ก็ต่อเครื่องที่เราใช้จะต้องมีโปรแกรม Web Server เช่น Apache 1.3.x นอกจากนั้นยังต้องการโปรแกรม PHP version 3 หรือที่สูงกว่านี้ ซึ่งการติดตั้งโปรแกรม Piranha Web Interface นั้นจะทำโดยการติดตั้งโปรแกรมที่มีอยู่ 3 ส่วนคือ

1 piranha

2 ipvsadm

3 piranha-gui

ซึ่งทั้ง 3 โปรแกรมนี้จะอยู่ในรูปแบบของ RPM package เพียง package เดียว

ในการติดตั้งโปรแกรม Piranha Web Interface นั้นเมื่อทำการติดตั้งเสร็จสมบูรณ์โปรแกรมการติดตั้งก็จะทำการสร้างชื่อผู้ใช้คนใหม่ที่ชื่อ piranha ภายในระบบของเครื่องที่ลงโปรแกรม ซึ่งเป็นบัญชีชื่อผู้ใช้ของระบบและจะทำให้เราไม่สามารถ login เข้ามาใช้เครื่องด้วยชื่อผู้ใช้ที่มีชื่อเป็น piranha ซึ่งชื่อนี้จะใช้สำหรับการใช้งานของ Piranha Web Interface เพียงอย่างเดียวเท่านั้น

ระบบโดยทั่วไปของโปรแกรม Piranha Web Interface

เมื่อเริ่มแรกที่จะใช้งานโปรแกรม Piranha Web Interface ได้นั้น เราจะต้องทำการสร้างรหัสผ่านให้กับผู้ใช้งานที่มีชื่อเป็น piranha ซึ่งเราจะทำได้ก็ต่อเมื่อเราทำการ login ที่มีชื่อผู้ใช้เป็น root เสียก่อนแล้วจึงค่อยใช้คำสั่ง

```
/usr/sbin/piranha-passwd <password>
```

จากคำสั่งข้างบน สัญลักษณ์ <password> จะเป็นรหัสผ่านที่จะต้องป้อนให้กับคำสั่ง piranha – passwd โดยถ้าระบบที่เราใช้ก่อนหน้านี้ได้ทำการติดตั้งโปรแกรม Piranha Web Interface ไปแล้วและเราได้ทำการติดตั้งซ้ำใหม่ รหัสผ่านเก่าที่เราใช้ในการติดตั้งครั้งแรกจะยังคงใช้ได้อยู่

ถ้าคำสั่ง piranha-passwd ใช้งานไม่ได้ให้ใช้คำสั่งต่อไปนี้

```
passwd piranha
htpasswd /home/httpd/html/piranha/secure/passwords piranha
```

โดยที่คำสั่งแรกจะเป็นการสร้างรหัสผ่านให้กับชื่อผู้ใช้ที่ชื่อ piranha และคำสั่งที่สองจะเป็นการสร้างรหัสผ่านในการใช้งาน Piranha Web Interface

ในการใช้งานโปรแกรม Piranha Web Interface นั้นจะต้องการ Web browser ในการทำงาน ซึ่งการใช้งานจะทำได้โดยการเปิด Web browser แล้วให้ใส่ URL เป็น

http://localhost/piranha/

เมื่อ Web browser ได้ทำการติดต่อกับโปรแกรม Piranha Web Interface แล้วก็จะขึ้นหน้าจอให้ผู้ใช้ทำการ login เพื่อทำการใช้งานโปรแกรม ซึ่งการ login นั้นที่ช่อง User ID ให้ใส่ชื่อเป็น "piranha" และใส่รหัสผ่านที่ช่อง Password โดยรหัสผ่านที่ใช้จะได้จากการกำหนดรหัสผ่านในคำสั่ง piranha-passwd หรือรหัสผ่านที่เราได้กำหนดผ่านทางคำสั่ง

htpasswd /home/httpd/html/piranha/secure/passwords piranha

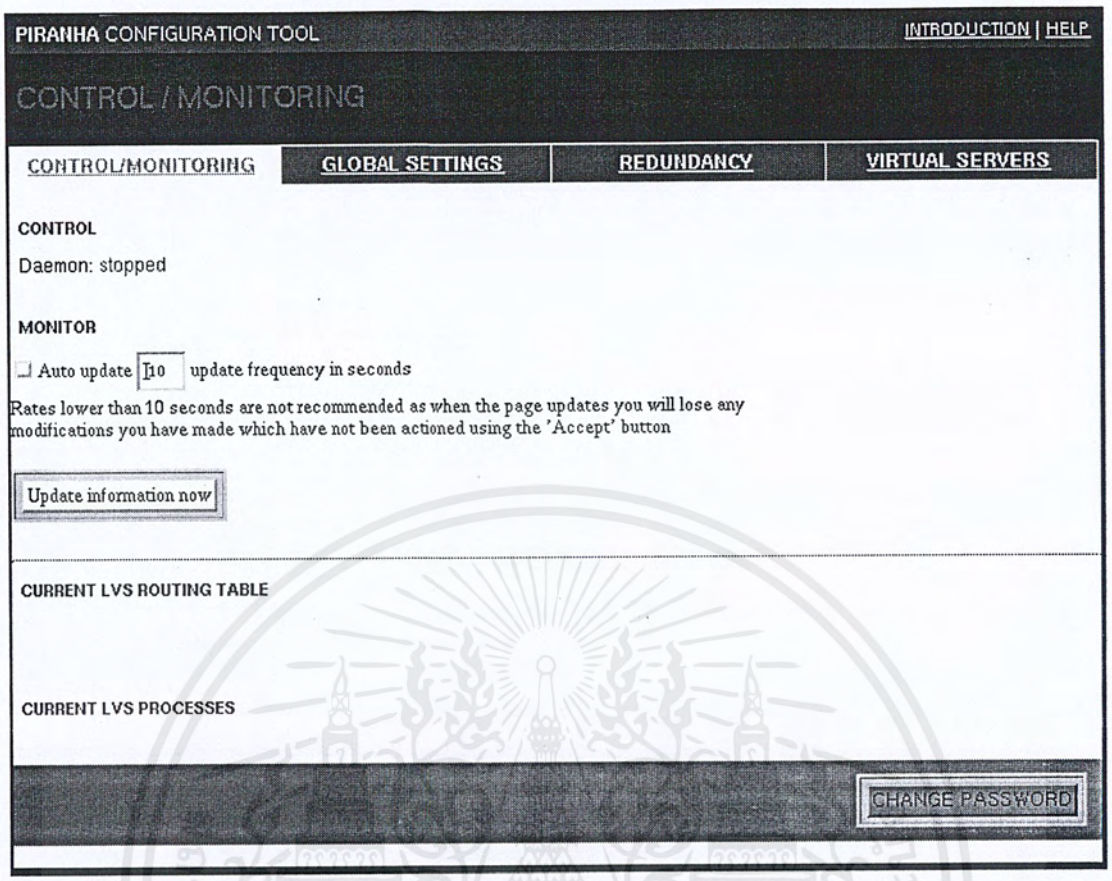
ในการใช้งาน Piranha Web Interface เพื่อทำการปรับแต่งระบบคลัสเตอร์นั้นจะต้องใช้วิธีการต่อไปนี้ในการปรับแต่งระบบ

- การเลือกของการปรับแต่งระบบจะใช้ radio buttons
- เมื่อได้ปรับแต่งระบบเรียบร้อยแล้วและต้องการเก็บข้อมูลจะต้องทำการกดปุ่ม ACCEPT ซึ่งถ้าไม่ได้ทำการกดปุ่ม ACCEPT และทำการเปลี่ยนหน้าจอใหม่ ข้อมูลที่ได้ปรับแต่งไปในหน้าจอนั้นก็จะหายไปยังคงไม่ได้ถูกเก็บลงในเครื่อง

การปรับแต่งระบบคลัสเตอร์ภายในโปรแกรม Piranha Web Interface นั้นจะมีอยู่หลายส่วนดังนี้

CONTROL/MONITORING Panel

เมื่อทำการ login เข้ามาใช้โปรแกรม Piranha Web Interface ได้แล้วนั้นหน้าจอแรกที่เห็นจะเป็นไปตามรูปที่ 4.1 หน้าจอของ CONTROL/MONITORING ซึ่งหน้าจอนี้จะใช้เป็นตัวดูสถานะการทำงานของระบบคลัสเตอร์ และยังสามารถที่จะกำหนดให้หน้าจอนี้ทำการปรับปรุงเปลี่ยนแปลงข้อมูลได้โดยอัตโนมัติ ซึ่งวิธีการที่จะทำให้การดูสถานะการทำงานของระบบคลัสเตอร์เป็นไปแบบอัตโนมัตินั้นจะต้องทำโดยการกดปุ่ม Auto update และใส่ตัวเลขที่แทนความถี่ในการเปลี่ยนแปลงหน้าจอ ซึ่งหมายความว่าต้องการให้หน้าจอนี้ทำการปรับปรุงเปลี่ยนแปลงข้อมูลที่หน้าจอนี้กี่วินาทีต่อครั้ง และจากนั้นก็กดปุ่ม ACCEPT เพื่อให้โปรแกรม Piranha Web Interface ทำงานตามที่เราได้กำหนดไว้

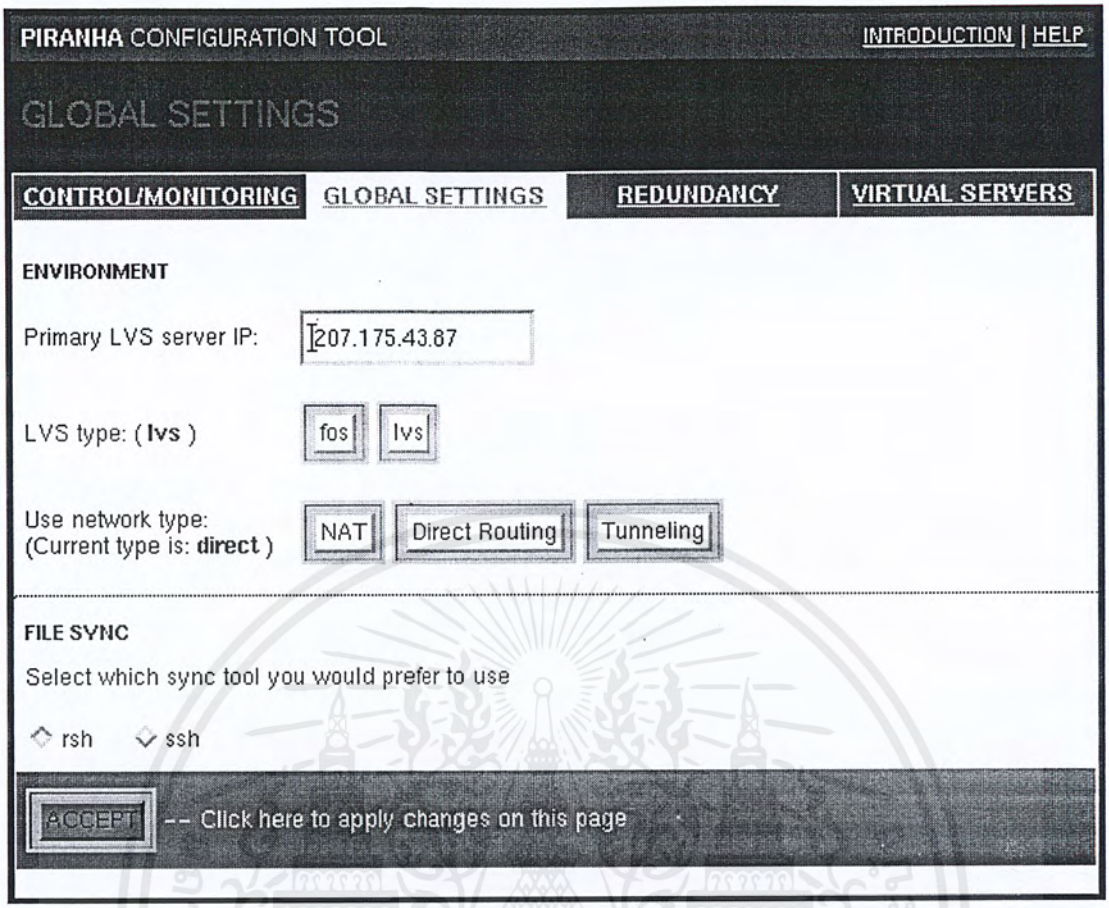


รูปที่ 4.1 แสดงหน้าจอของ CONTROL/MONITORING Panel

จากรูปที่ 4.1 เรายังสามารถทำการปรับปรุงเปลี่ยนแปลงข้อมูลที่แสดงสถานะการทำงานของคลัสเตอร์ในขณะใดขณะหนึ่งได้โดยการกดปุ่ม Update information now

GLOBAL SETTINGS Panel

เมื่อเราได้ทำการคลิกที่แท็ป GLOBAL SETTILNGS หน้าจอจะเปลี่ยนไปตามรูปที่ 4.2



รูปที่ 4.2 แสดงหน้าจอของ GLOBAL SETTINGS Panel

จากหน้าจอรูปที่ 4.2 นี้เราจะสามารถกำหนดรูปแบบของการคลัสเตอร์ที่เราจะนำมาใช้ในการทำงานได้ ค่าแรกที่ต้องทำการใส่คือหมายเลขไอพีของเครื่องโหลดบาลานซ์ที่จะทำการรับการร้องขอบริการจากอินเตอร์เน็ตโดยจะใส่ในช่อง Primary LVS server IP และจากนั้นจะทำการเลือกชนิดของคลัสเตอร์ที่จะนำมาใช้งานซึ่งมีอยู่ 2 ชนิดคือ

- LVS (Linux Virtual Services)
- FOS (Fail Over Services)

ถ้าเราทำการเลือกเป็นชนิด LVS ก็จะมีทางเลือกให้เราเลือกว่าจะทำการติดตั้ง LVS โดยใช้วิธีแบบไหนในการทำงาน ซึ่งมีอยู่ทั้งหมด 3 แบบคือ

- NAT (Network Address Translation)
- Direct Routing
- Tunneling

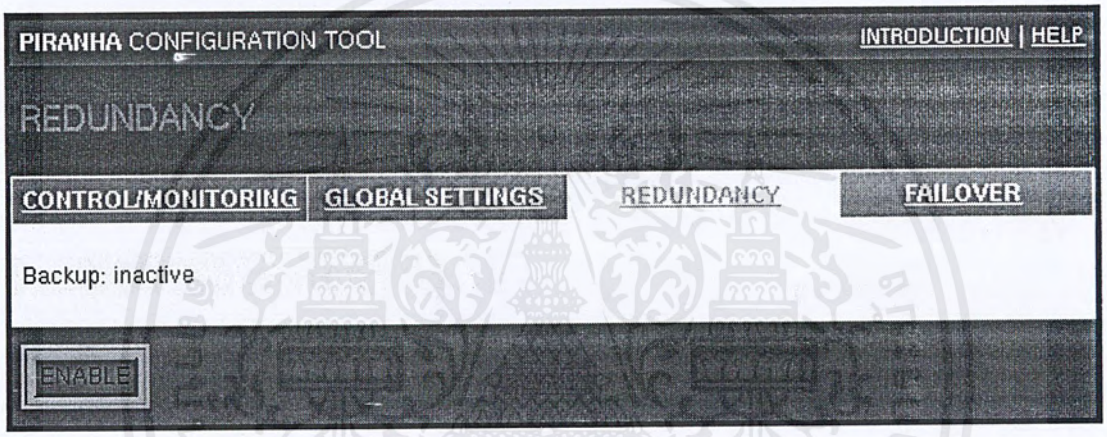
ถ้าเราทำการเลือกวิธีการติดตั้งของ LVS เป็น NAT ก็จะมี 2 ค่าที่เราจะต้องทำการใส่ ซึ่งมีดังนี้

- The NAT router IP address
- The router device

ต่อจากนั้นจะเป็นการเลือกวิธีการที่จะใช้เพื่อทำการติดต่อกันของเครื่องต่างๆภายในคลัสเตอร์ โดยค่าตั้งต้นที่โปรแกรมกำหนดมาให้จะเป็น rsh ซึ่งเราก็สามารถใช้ ssh ในการทำงานได้โดยที่ระบบของเราจะต้องมีโปรแกรมนี้ติดตั้งอยู่ในระบบแล้ว

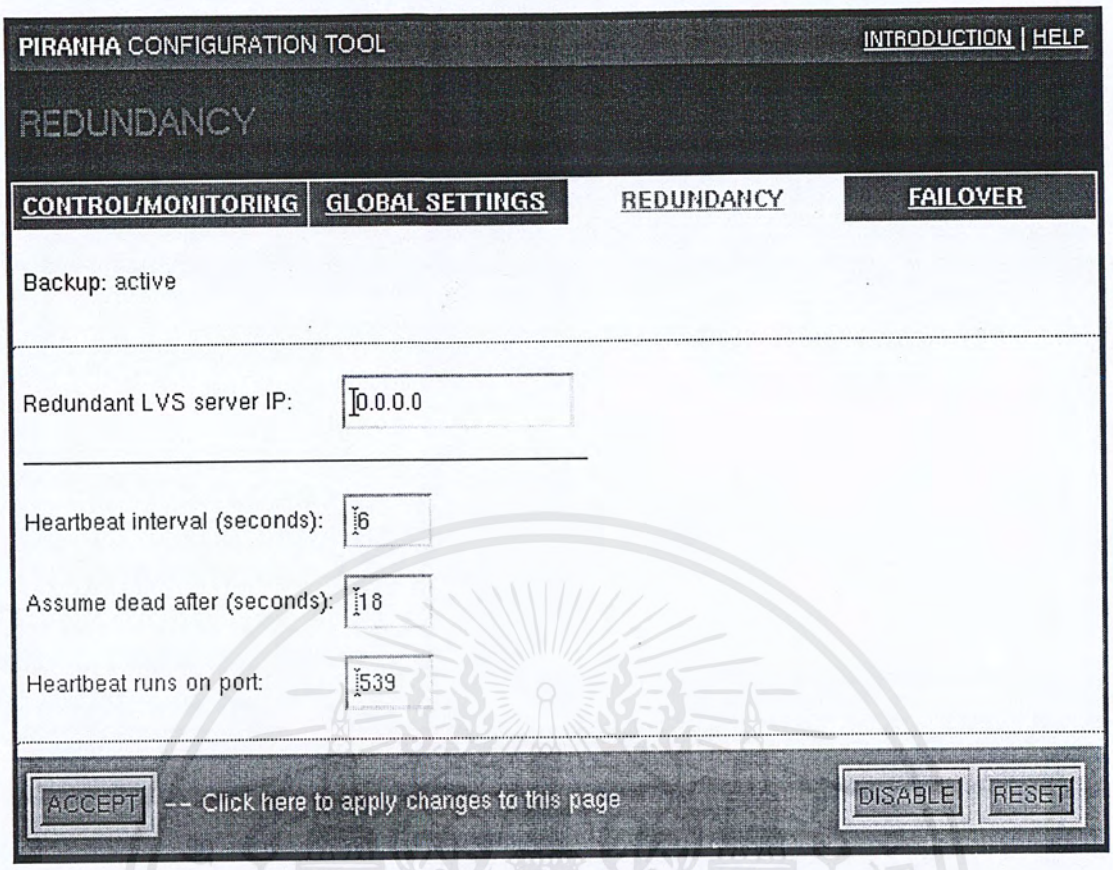
REDUNDANCY Panel

จากนั้นเมื่อคลิกเลือกที่หัวข้อ REDUNDANCY เราจะเห็นหน้าจอขึ้นมาตามรูปที่ 4.3 เพื่อให้เราทำการใส่ข้อมูลตามที่เราต้องการติดตั้งภายในระบบคลัสเตอร์ของเรา โดยค่าเริ่มแรกของหน้าจอนี้จะไม่มีข้อมูลอยู่เลย เพราะว่าเรายังไม่ได้ทำการติดตั้ง



รูปที่ 4.3 แสดงหน้าจอเริ่มแรกของ Redundancy Panel

เมื่อเราต้องการติดตั้งแบบ REDUNDANCY ก็ให้ทำการกดปุ่ม ENABLE และจากนั้นหน้าจอก็จะเปลี่ยนเป็นตามรูปที่ 4.4

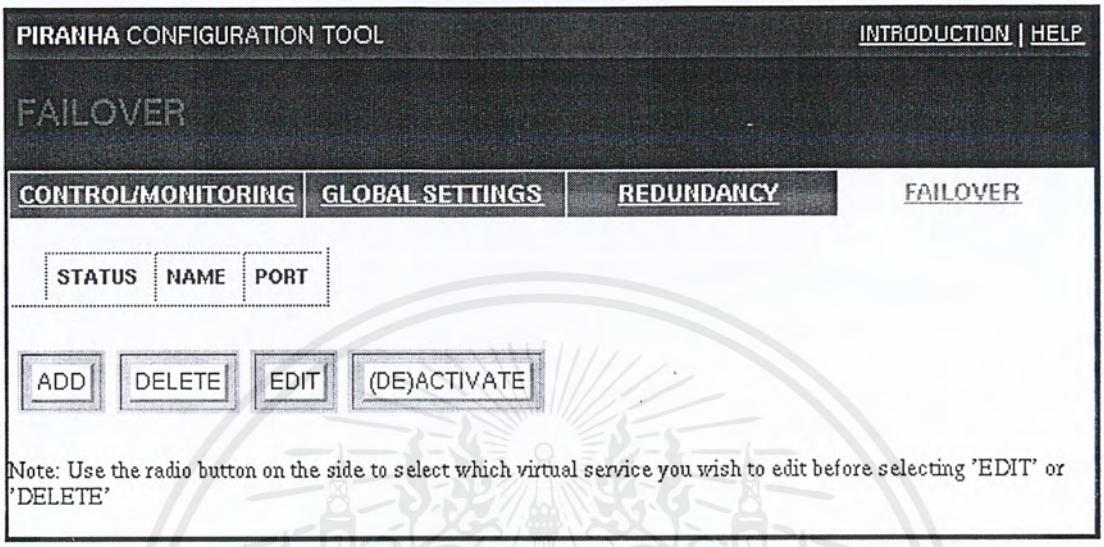


รูปที่ 4.4 แสดงหน้าจอ Redundancy Panel ที่ใช้ในการติดตั้ง

จากนั้นก็ทำการแก้ไขข้อมูลให้เป็นไปตามที่เราต้องการใช้ในการติดตั้ง โดยค่าที่ต้องใส่ให้กับหน้าจอนี้คือค่าในช่อง Redundant LVS server IP ซึ่งตัวเลขนี้จะเป็นหมายเลขไอพีของเครื่องที่จะทำหน้าที่ในการ Redundancy ส่วนช่องที่ต้องใส่ที่เหลือจะมีการใส่ค่าเริ่มแรกที่ระบบกำหนดมาให้ โดยปกติแล้วค่าเหล่านี้จะเป็นค่าที่ระบบสามารถทำงานได้

FAILOVER Panel

ในส่วนของ GLOBAL SETTINGS Panel นั้นถ้าเราได้ทำการติดตั้งเป็นแบบ FOS แล้วในหัวข้อแท็บที่ 4 จะเป็นหัวข้อ FAILOVER ซึ่งเมื่อเราทำการคลิกเลือกหัวข้อเข้าไปจะทำให้มีหน้าจอที่แสดงออกมาเป็นไปตามรูปที่ 4.5

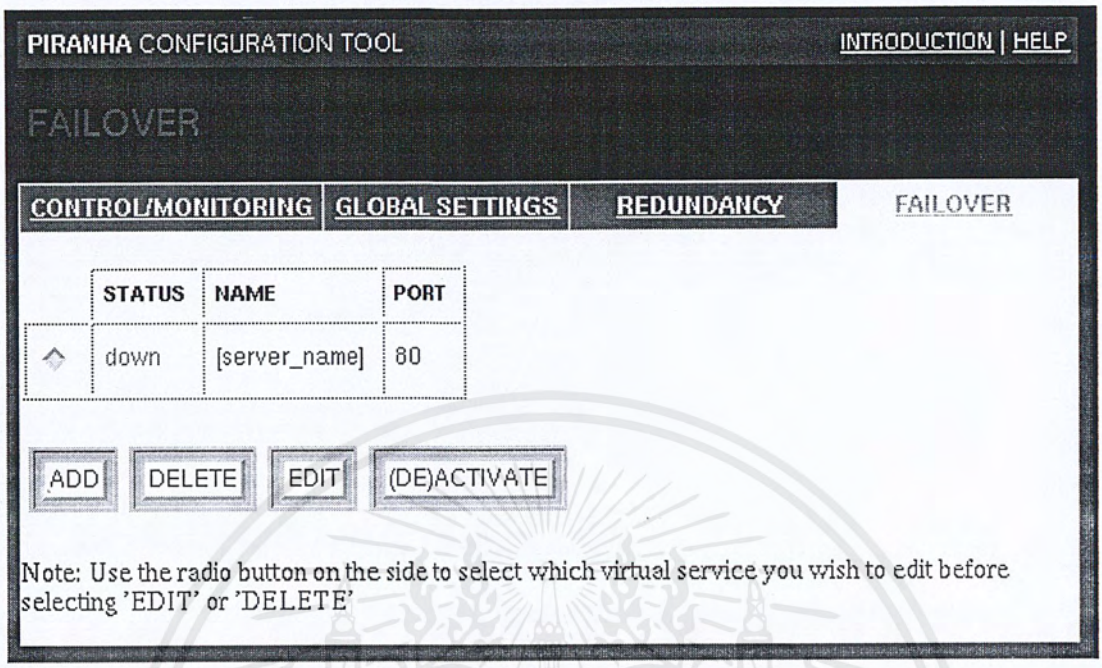


รูปที่ 4.5 แสดงหน้าจอเริ่มแรกของ Failover Panel

ซึ่งที่หน้าจอตามรูปที่ 4.5 จะมีปุ่มต่อไปนี้ที่ใช้ในการติดตั้ง

- ADD ทำการสร้างหัวข้อเรื่องที่จะใช้ในการติดตั้ง
- DELETE ทำการลบหัวข้อเรื่องที่เราต้องการจะลบ
- EDIT ทำการแก้ไขหัวข้อเลือกที่เราได้เลือกไว้
- (DE)ACTIVATE ทำให้หัวข้อที่เราเลือกไว้ทำงานหรือว่าจะไม่ให้มันทำงาน

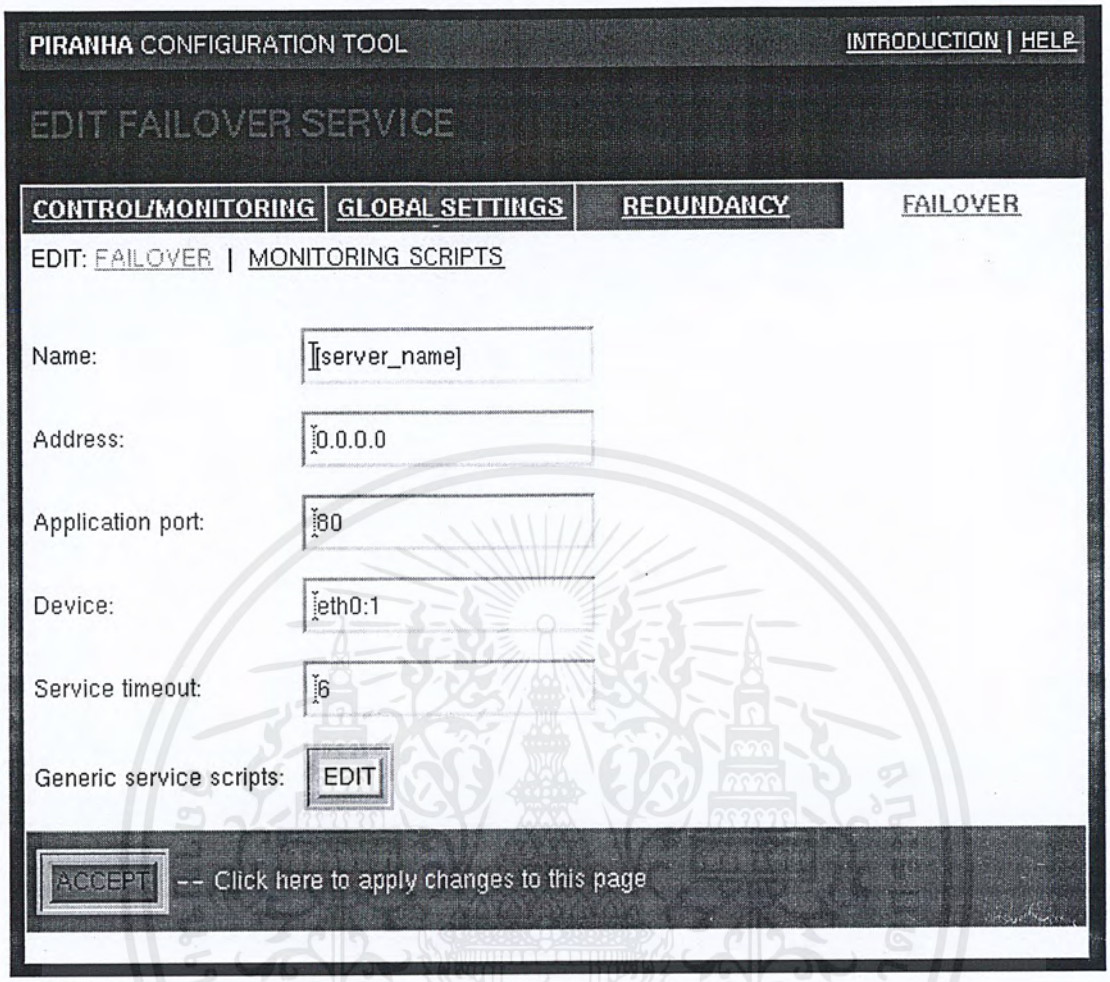
ทำการกดที่ปุ่ม ADD เพื่อเริ่มทำการติดตั้งเซิร์ฟเวอร์ที่จะใช้ในการทำงานของระบบคลัสเตอร์ และที่หน้าจอ FAILOVER จะมีหัวข้อของเซิร์ฟเวอร์ใหม่ที่เราจะทำการติดตั้ง ซึ่งหน้าจอจะเป็นไปตามรูปที่ 4.6



รูปที่ 4.6 แสดงหน้าจอของ Failover Panel ที่ได้ทำการติดตั้ง

จากรูปที่ 4.6 ถ้าเราทำการเพิ่มเครื่องเซิร์ฟเวอร์มากกว่าหนึ่งเครื่องภายในระบบ เวลาจะทำการปรับปรุงเปลี่ยนแปลงข้อมูลของเซิร์ฟเวอร์เครื่องใดให้ทำการกดปุ่มที่ radio button ที่อยู่ข้างหน้าชื่อของเซิร์ฟเวอร์ก่อนแล้วจึงค่อยทำการเปลี่ยนแปลงข้อมูล

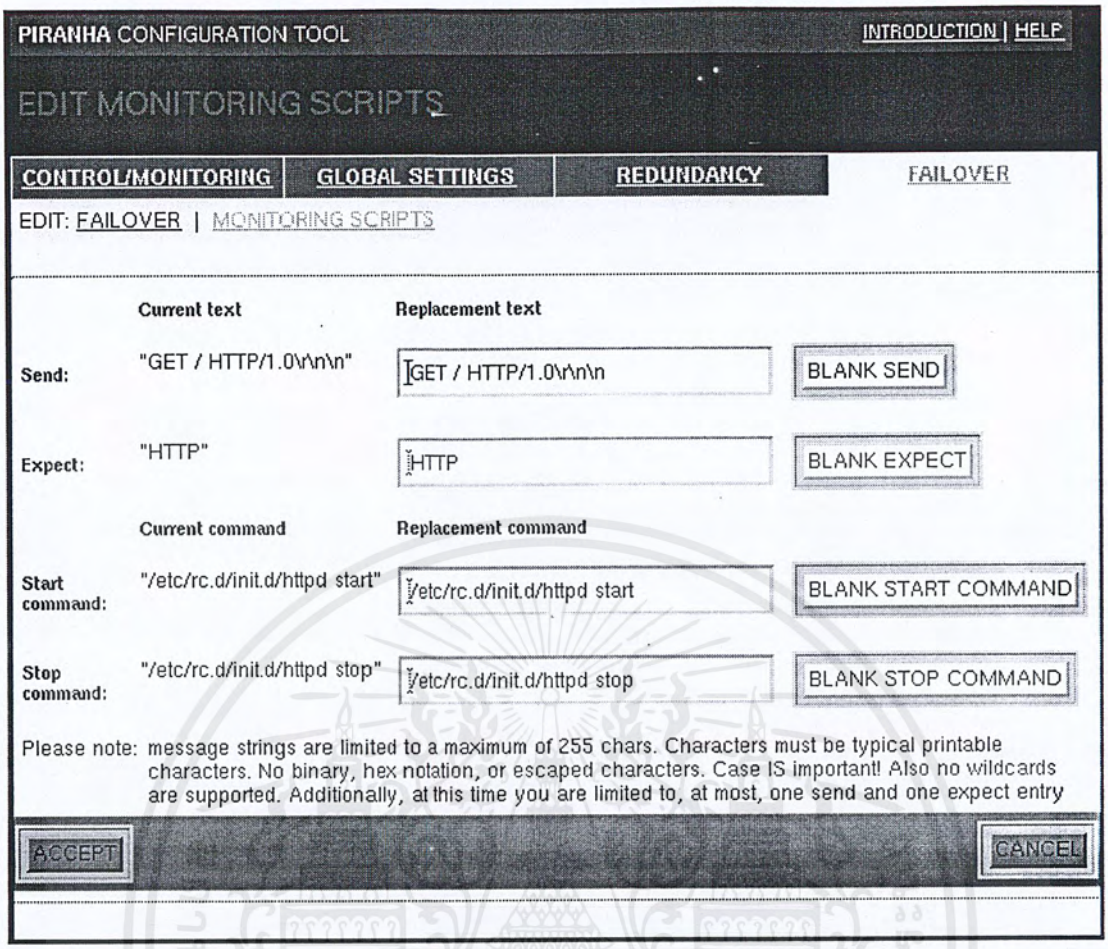
จากนั้นเมื่อเราทำการกดปุ่ม EDIT ก็จะมีหน้าจอตามรูปที่ 4.7 ปรากฏขึ้นมา.



รูปที่ 4.7 แสดงหน้าจอ Failover Panel ที่ได้ทำการแก้ไข

จากรูปที่ 4.7 หน้าจอนี้จะมีช่องของ Name กับช่องของ Address มาให้ใส่ค่าที่เราต้องการติดตั้ง ซึ่งจะมีค่าเริ่มแรกที่โปรแกรมให้มา ซึ่งเราจำเป็นต้องทำการแก้ไขให้ค่าเหล่านี้เป็นไปตามที่เราทำการติดตั้งภายในระบบคลัสเตอร์ของเรา

จากหน้าจอตามรูปที่ 4.7 เมื่อเราทำการติดตั้งให้เป็นไปตามที่ระบบคลัสเตอร์ของเราทำงานเสร็จแล้ว ก็ให้ทำการกดปุ่ม ACCEPT เพื่อทำการบันทึกค่าที่เราได้ทำการติดตั้งไว้ จากนั้นทำการกดปุ่ม EDIT ซึ่งจะปรากฏหน้าจอที่มีลักษณะตามรูปที่ 4.8

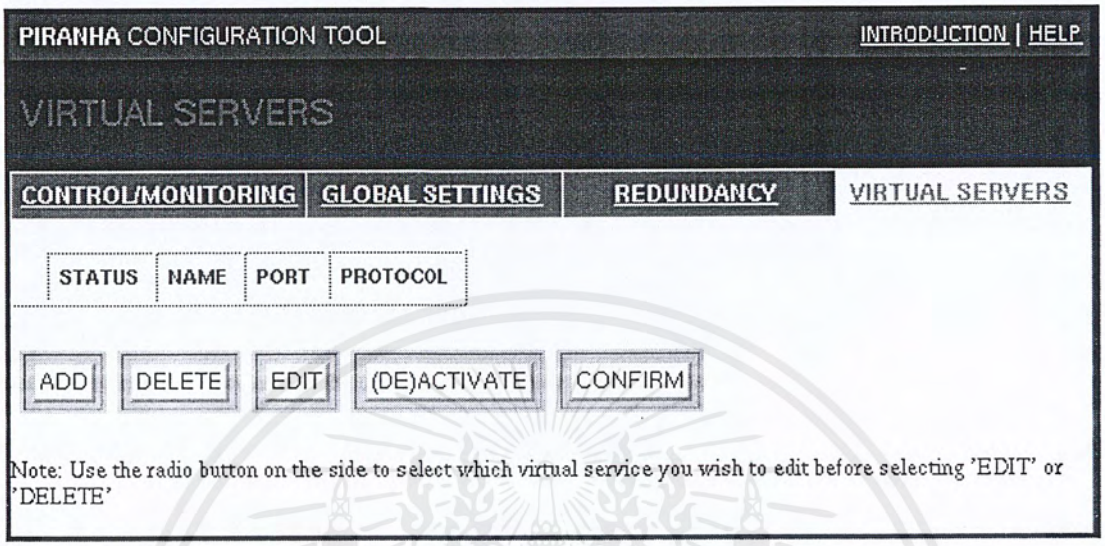


รูปที่ 4.8 แสดงหน้าจอ Failover Panel ที่ใช้แก้ไข script

จากรูปที่ 4.8 script ที่ติดตั้งมาให้โดยโปรแกรมนั้นสามารถทำงานได้เลยโดยที่ไม่ต้องทำการแก้ไข ซึ่งถ้าเราต้องการแก้ไขเพื่อใช้ในการทำงานของระบบคลัสเตอร์ของเราก็สามารถทำได้

VIRTUAL SERVERS Panel

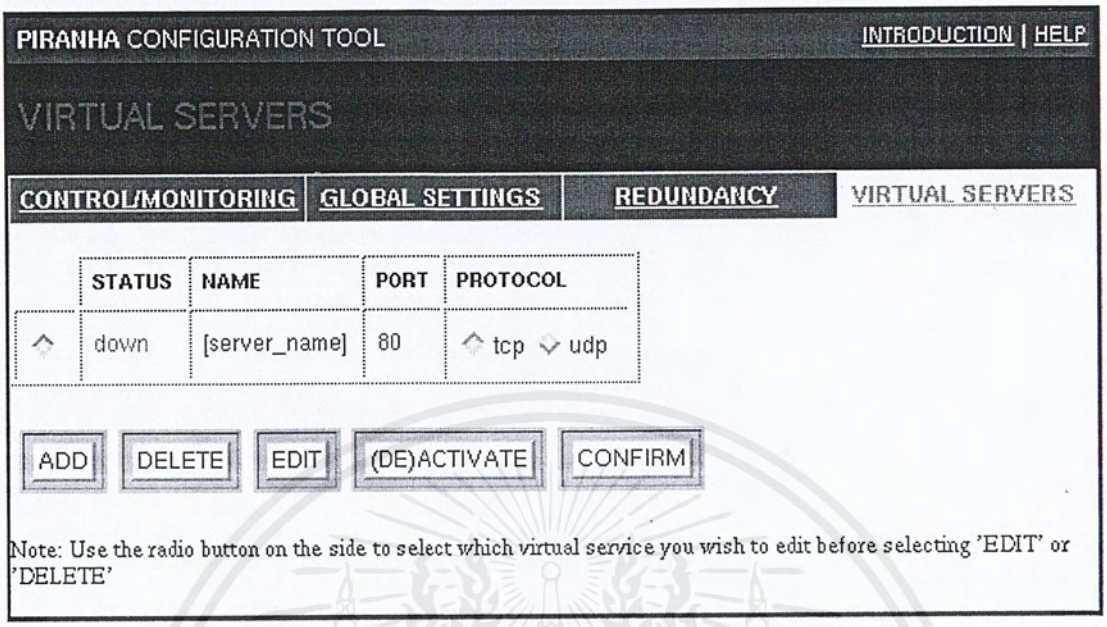
ในส่วนของ GLOBAL SETTINGS Panel นั้นถ้าเราได้ทำการติดตั้งเป็นแบบ LVS แล้วในหัวข้อแท็ปที่ 4 จะเป็นหัวข้อ VIRTUAL SERVERS ซึ่งเมื่อเราทำการคลิกเลือกหัวข้อเข้าไปจะทำให้มีหน้าจอที่แสดงออกมาเป็นไปตามรูปที่ 4.9



รูปที่ 4.9 แสดงหน้าจอเริ่มแรกของ VIRTUAL SERVERS Panel

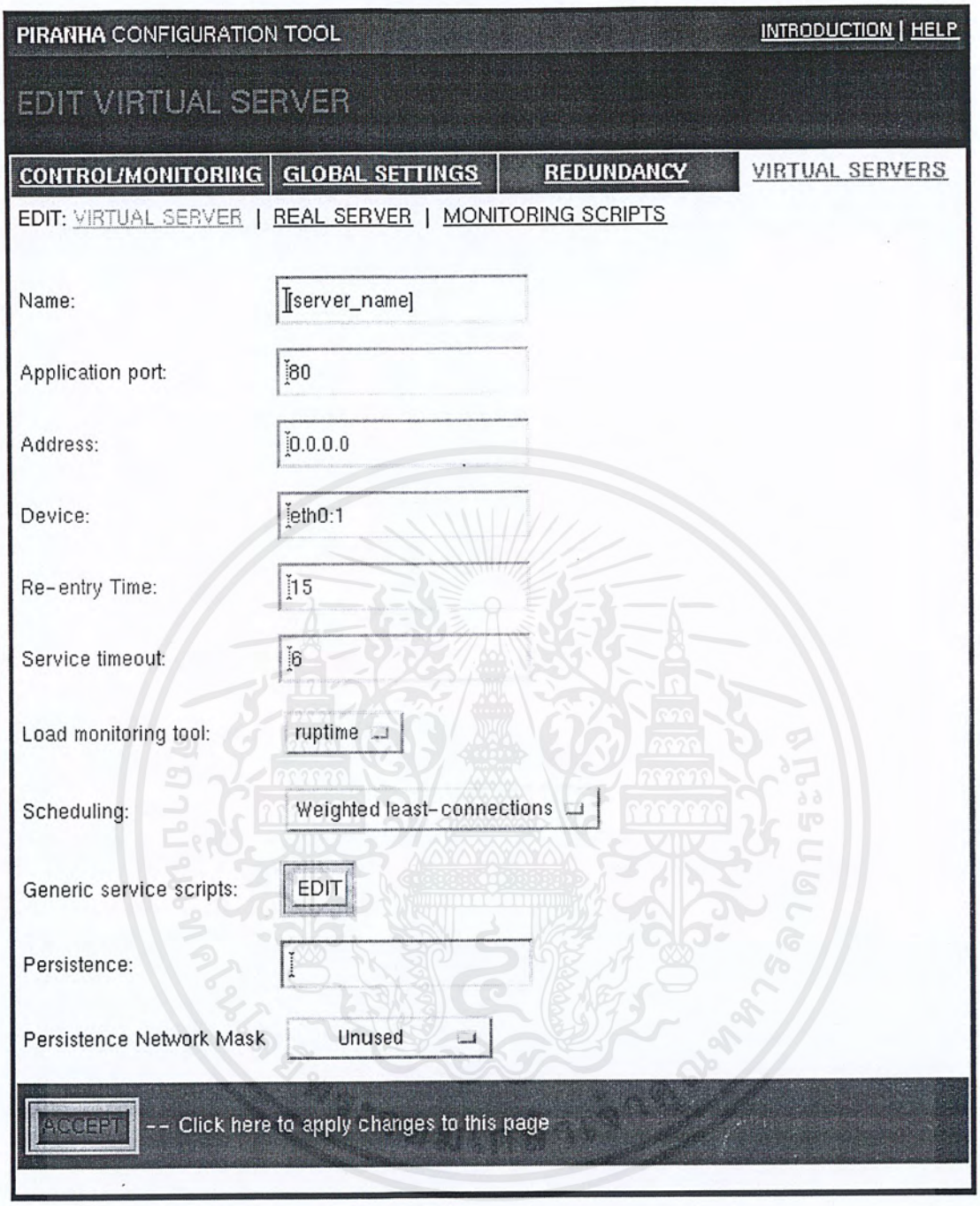
ตามรูปที่ 4.9 ที่ช่อง PROTOCOL จะมีค่าที่เราใช้ได้อยู่สองอย่าง คือแบบ TCP และแบบ UDP ซึ่งปุ่ม CONFIRM จะใช้สำหรับการยืนยันกับ PROTOCOL ที่เราได้ทำการเลือกใช้ในระบบของเรา

เมื่อเราต้องการเพิ่มเครื่องเซิร์ฟเวอร์ที่ใช้ในระบบคลัสเตอร์ของเราก็ให้ทำการกดปุ่ม ADD ซึ่งจะปรากฏหน้าจอตามรูปที่ 4.10



รูปที่ 4.10 แสดงหน้าจอ VIRTUAL SERVERS Panel ที่ได้ทำการเพิ่มเครื่องเซิร์ฟเวอร์

เมื่อเครื่องเซิร์ฟเวอร์ใหม่ได้รับกรเพิ่มเข้าไป ทำการเลือกเครื่องเซิร์ฟเวอร์ที่ต้องการเปลี่ยนแปลงค่า โดยการกดปุ่ม radio button ที่อยู่ข้างหน้าชื่อเครื่องเซิร์ฟเวอร์ แล้วทำการกดปุ่ม EDIT จากนั้นจะปรากฏหน้าจอเป็นไปตามรูปที่ 4.11



รูปที่ 4.11 แสดงหน้าจอ VIRTUAL SERVERS Panel ที่ทำการแก้ไขค่าติดตั้งของเครื่องเซิร์ฟเวอร์

ตามรูปที่ 4.11 หน้าจอนี้จะมีค่าเริ่มแรกที่โปรแกรมได้ทำการติดตั้ง ซึ่งที่ช่อง Name กับ Address เราจำเป็นจะต้องทำการใส่ค่าที่เราใช้งานภายในระบบเพื่อให้ระบบคลัสเตอร์ของเราทำงานได้ ส่วนค่าอื่นๆก็ให้ทำการปรับเปลี่ยนแก้ไขค่าให้เป็นไปตามความต้องการใช้งานของระบบของเรา

บทที่ 5

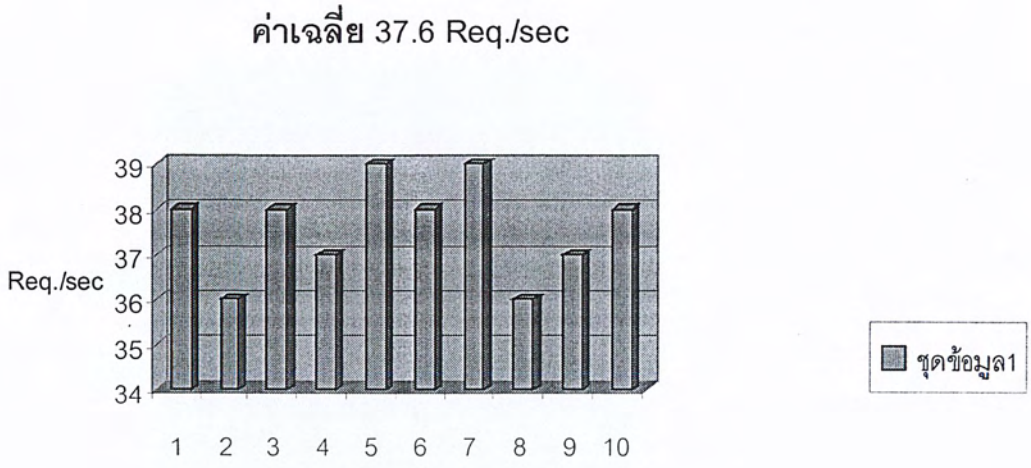
ผลการทดสอบ

ภายในโครงงานนี้ต้องการเปรียบเทียบประสิทธิภาพของเว็บเซิร์ฟเวอร์ เมื่อใช้ระบบ Linux Clustering แบบต่างๆ โดยจะเน้นเว็บเซิร์ฟเวอร์ที่ใช้วิธีการของ Linux virtual server เป็นหลัก และโปรแกรมที่ใช้ในการทดสอบคือ โปรแกรม Webbench นั่นเอง

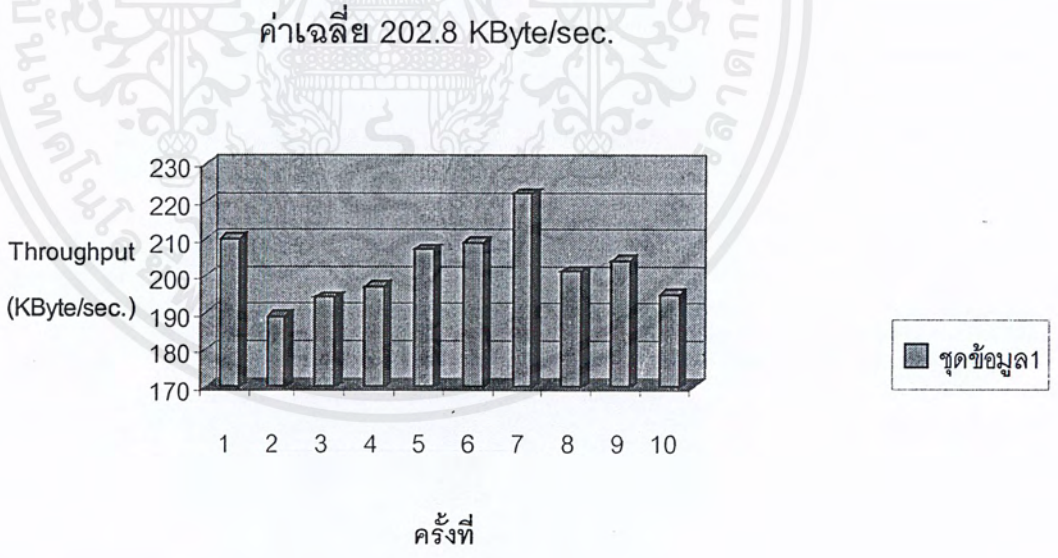
ในการทดสอบใดๆก็ตามจำเป็นต้องมีสิ่งที่ใช้อ้างอิง ในการทดสอบนี้ก็เช่นกัน โดยในสองรูปแรกของบทนี้จะเป็นผลการทดสอบจากเว็บเซิร์ฟเวอร์ปกติเพียงเครื่องเดียว ผลที่แสดงออกมาจะประกอบไปด้วยจำนวน Request ที่สามารถรองรับได้ในหนึ่งวินาที (Req./sec.) และอีกรูปหนึ่งจะแสดงอัตรา Throughput ของเว็บเซิร์ฟเวอร์ (Kbyte/sec.)

ในรูปถัดๆมาจะเป็นผลการทดสอบเมื่อใช้วิธีการของ Linux virtual server โดยจะเริ่มจากการมี real server เพียงหนึ่งเครื่อง แล้วทำการเพิ่มเป็น สองเครื่อง และสามเครื่องตามลำดับ

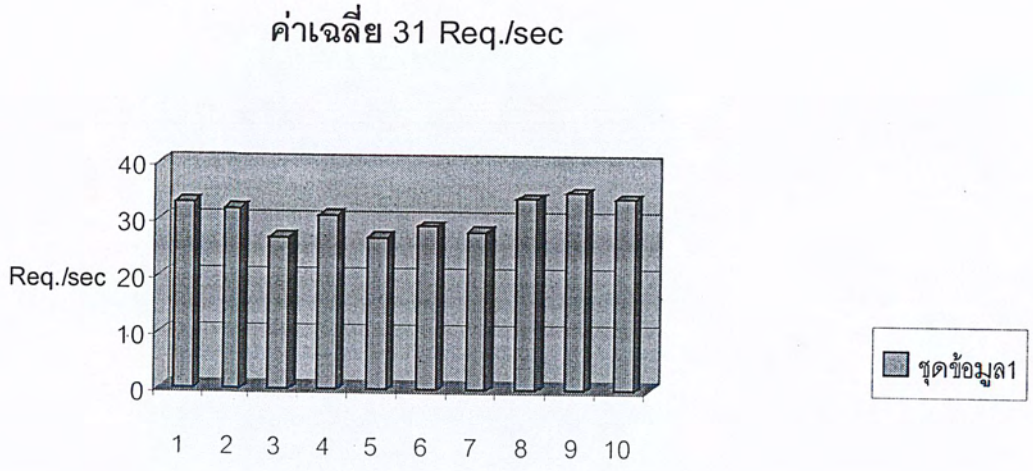
การทดสอบโดยการเพิ่มจำนวนเครื่องทีละเครื่องนั้นเราใช้อัลกอริทึม Weighted Least Connection ในการทดสอบ ต่อมาจึงได้ทำการทดสอบกับอัลกอริทึมอื่นๆบ้าง อันได้แก่ Round Robin และ Weighted Round Robin



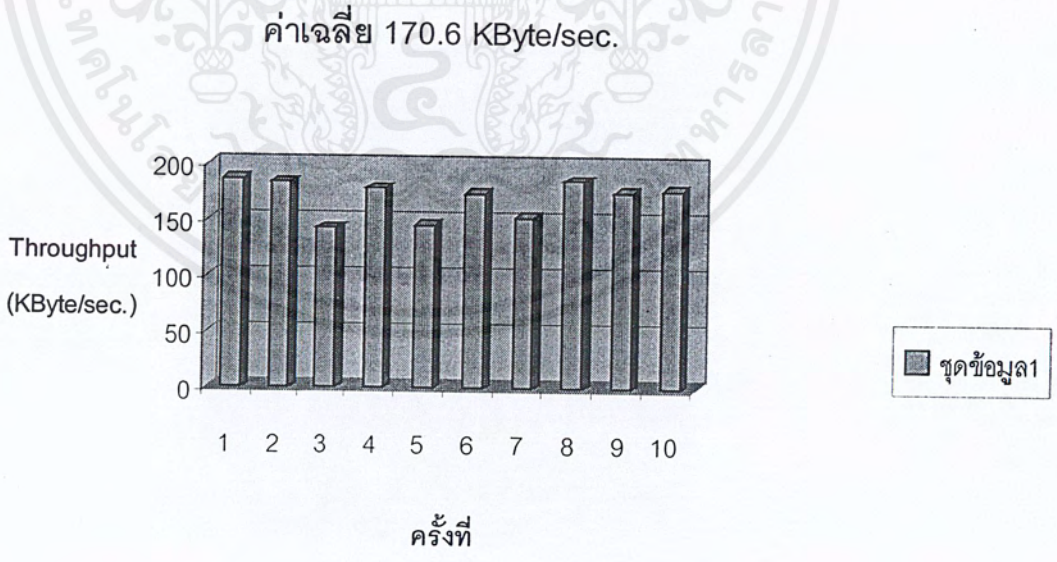
รูปที่ 5.1 แสดงค่า Req./sec ของเว็บเซิร์ฟเวอร์ 1 เครื่อง



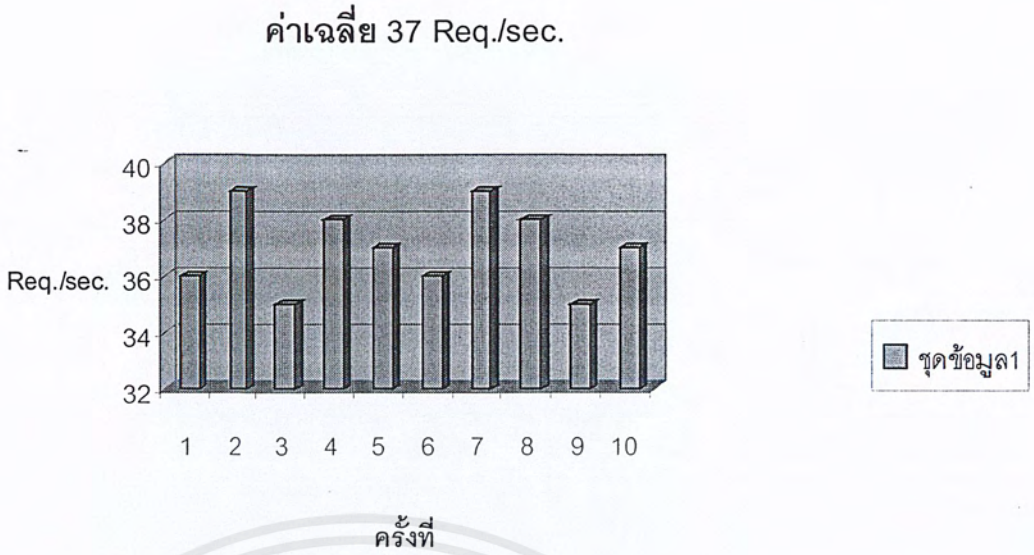
รูปที่ 5.2 แสดงค่า Throughput ของเว็บเซิร์ฟเวอร์ 1 เครื่อง



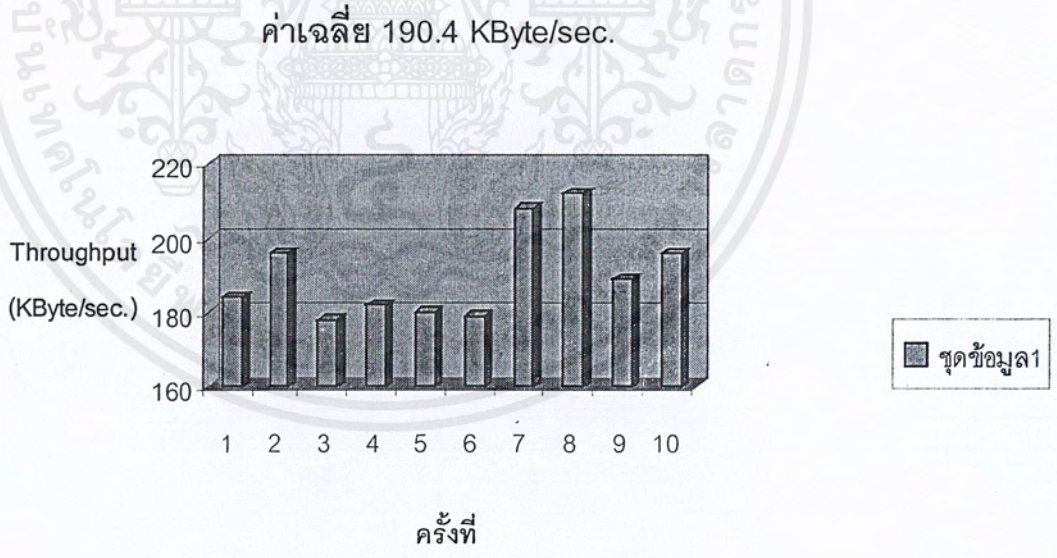
รูปที่ 5.3 แสดงค่า Req./sec ของ LVS – wlc 1 เครื่อง



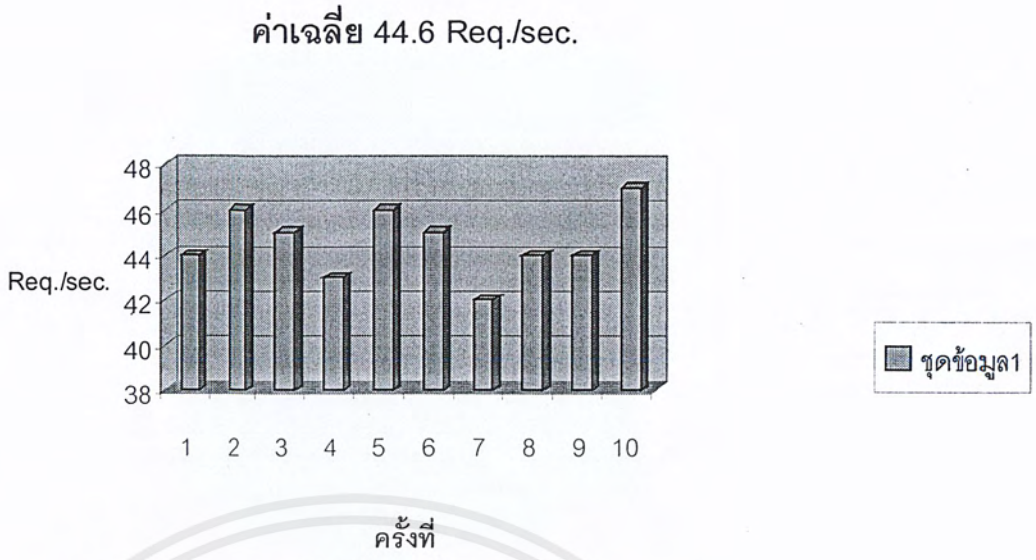
รูปที่ 5.4 แสดงค่า Throughput ของ LVS – wlc 1 เครื่อง



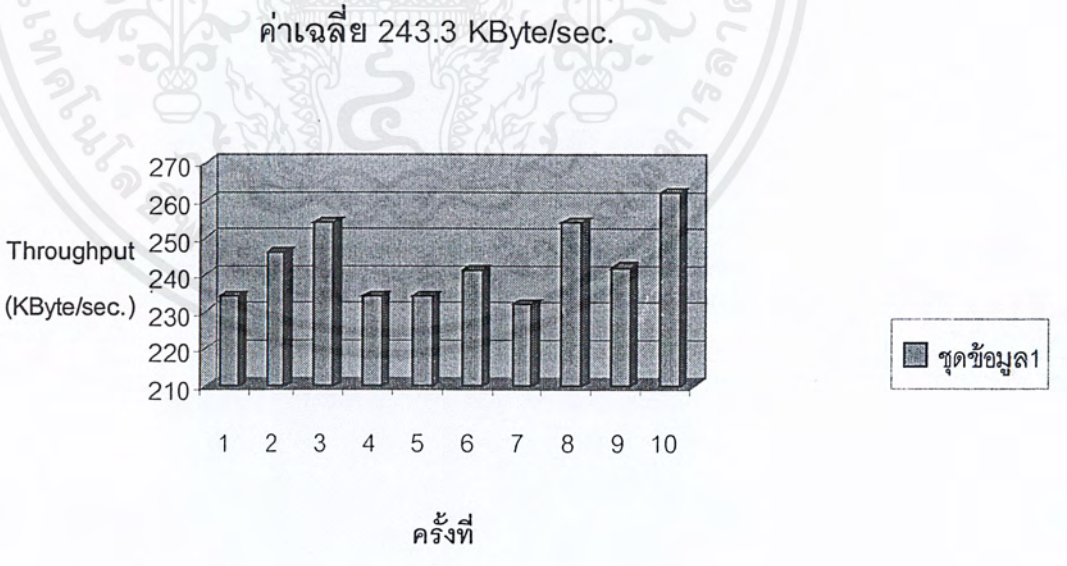
รูปที่ 5.5 แสดงค่า Req./sec. ของ LVS – wlc 2 เครื่อง



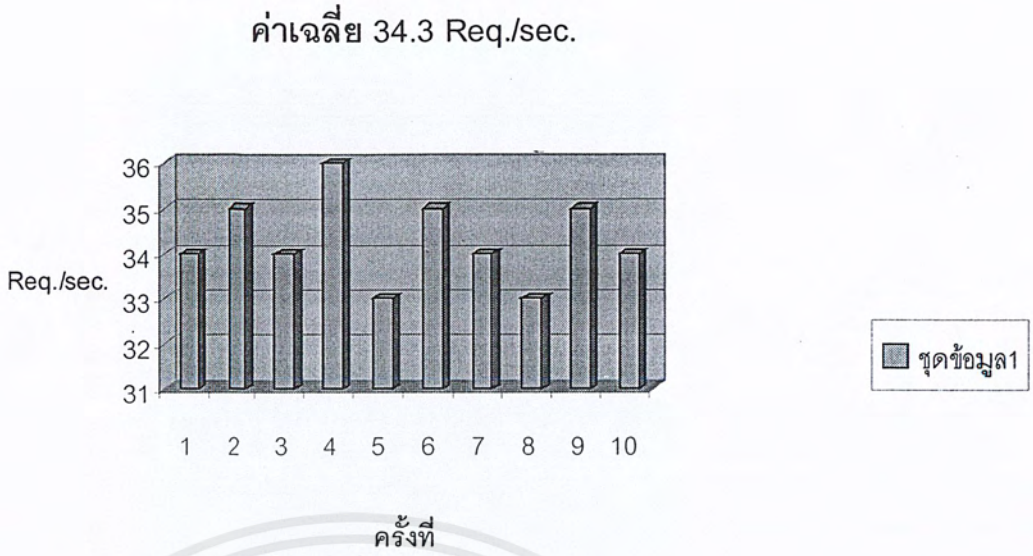
รูปที่ 5.6 แสดงค่า Throughput ของ LVS – wlc 2 เครื่อง



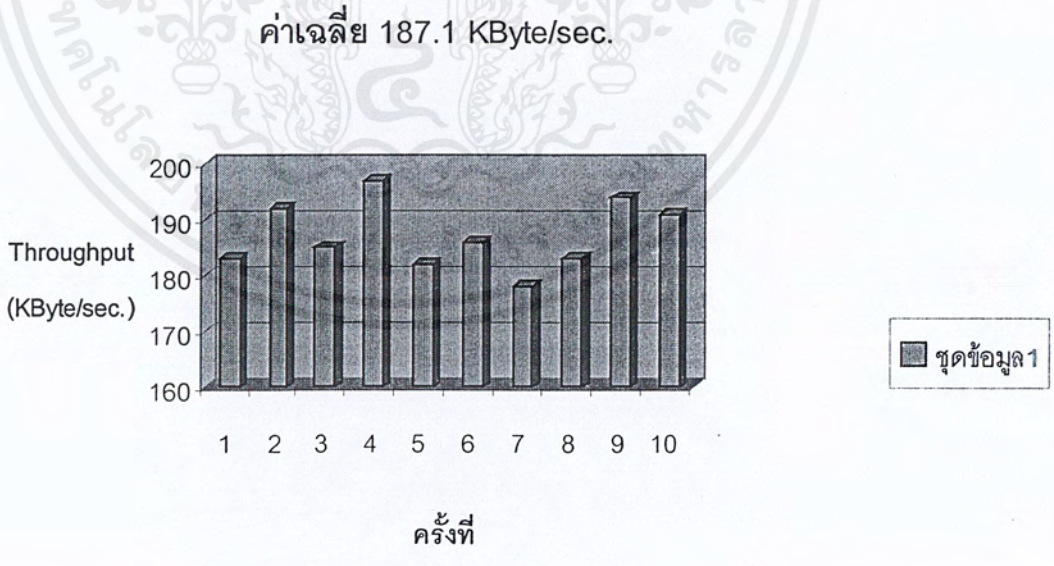
รูปที่ 5.7 แสดงค่า Req./sec. ของ LVS – wlc 3 เครื่อง



รูปที่ 5.8 แสดงค่า Throughput ของ LVS – wlc 3 เครื่อง

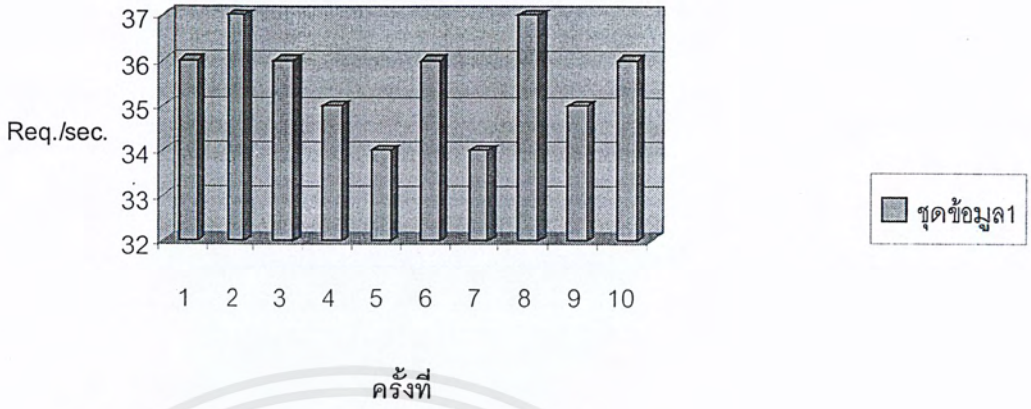


รูปที่ 5.9 แสดงค่า Req./sec. ของ LVS – rr 3 เครื่อง



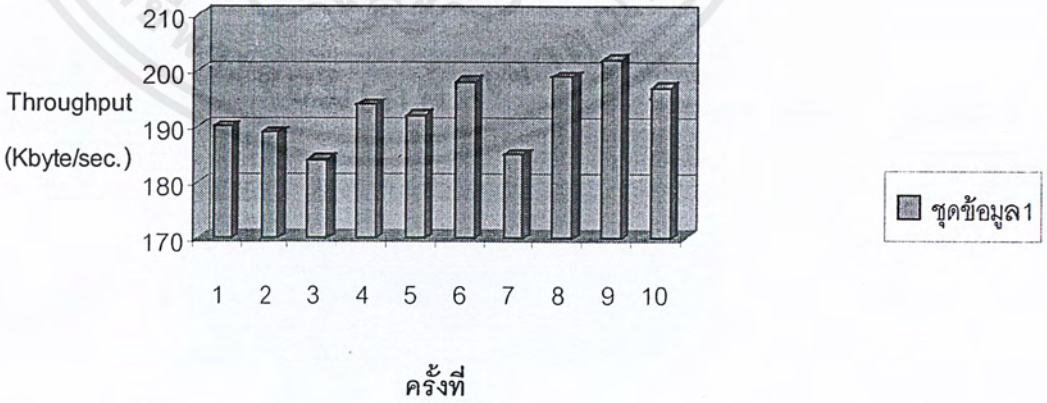
รูปที่ 5.10 แสดงค่า Throughput ของ LVS – rr 3 เครื่อง

ค่าเฉลี่ย 35.6 Req./sec.



รูปที่ 5.11 แสดงค่า Req./sec. ของ LVS – wrr 3 เครื่อง

ค่าเฉลี่ย 193 KByte/sec.



รูปที่ 5.12 แสดงค่า Throughput ของ LVS – wrr 3 เครื่อง

บทที่ 6
บทวิจารณ์และสรุปผล

6.1 สรุปผลการทดสอบ

จากผลการทดสอบในบทที่ผ่านมาจะได้ค่าเฉลี่ยต่างๆเป็นดังตารางที่ 6.1

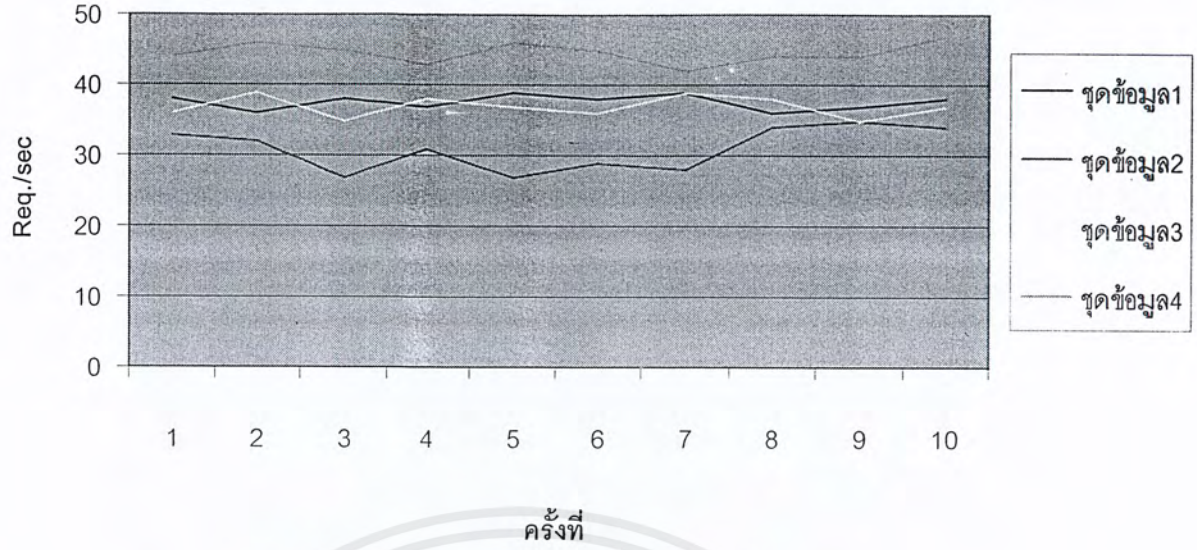
ชนิดของเว็บเซิร์ฟเวอร์	จำนวน Req./sec.	อัตรา Throughput (Kbyte/sec.)
เว็บเซิร์ฟเวอร์ 1 เครื่อง	37.6	202.8
LVS – wlc 1 เครื่อง	31	170.6
LVS – wlc 2 เครื่อง	37	190.4
LVS – wlc 3 เครื่อง	44.6	243.3

ตารางที่ 6.1 แสดงค่าเฉลี่ยของจำนวน Req./sec. และอัตรา Throughput เมื่อเพิ่ม Real server

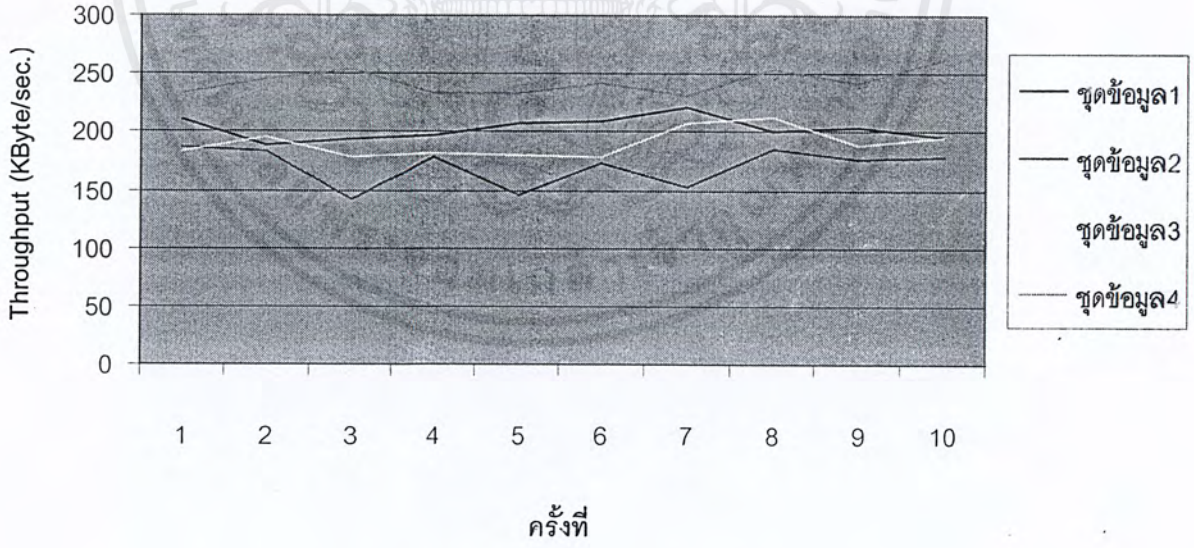
จากตารางจะเห็นได้ว่าเว็บเซิร์ฟเวอร์แบบ Linux virtual server ที่มี Real server เพียงเครื่องเดียวนั้นจะมีประสิทธิภาพด้อยกว่าเว็บเซิร์ฟเวอร์หนึ่งเครื่องธรรมดาเสียอีก สาเหตุก็เนื่องมาจากการส่งผ่านข้อมูลต่างๆนั้น จะต้องผ่านตัว Load balancer ก่อนเสมอ มีผลทำให้จำนวน Req./sec. และ อัตรา Throughput มีค่าลดลงอย่างเห็นได้ชัด

แต่เมื่อทำการเพิ่มจำนวน Real server ให้มากขึ้น มีผลให้ได้ประสิทธิภาพที่ดีขึ้นเรื่อยๆ โดยการที่มี Real server 3 เครื่อง จะเห็นได้ชัดถึงประสิทธิภาพที่เพิ่มขึ้น

รูปที่ 6.1 และ 6.2 จะแสดงการเปรียบเทียบให้เห็นถึงประสิทธิภาพที่เพิ่มขึ้นเมื่อทำการเพิ่ม Real server โดยจะมีเว็บเซิร์ฟเวอร์ปกติเป็นตัวอ้างอิง



รูปที่ 6.1 แสดงการเปรียบเทียบจำนวน Req./sec. เมื่อเพิ่มจำนวน Real server



รูปที่ 6.2 แสดงการเปรียบเทียบอัตรา Throughput เมื่อเพิ่มจำนวน Real server

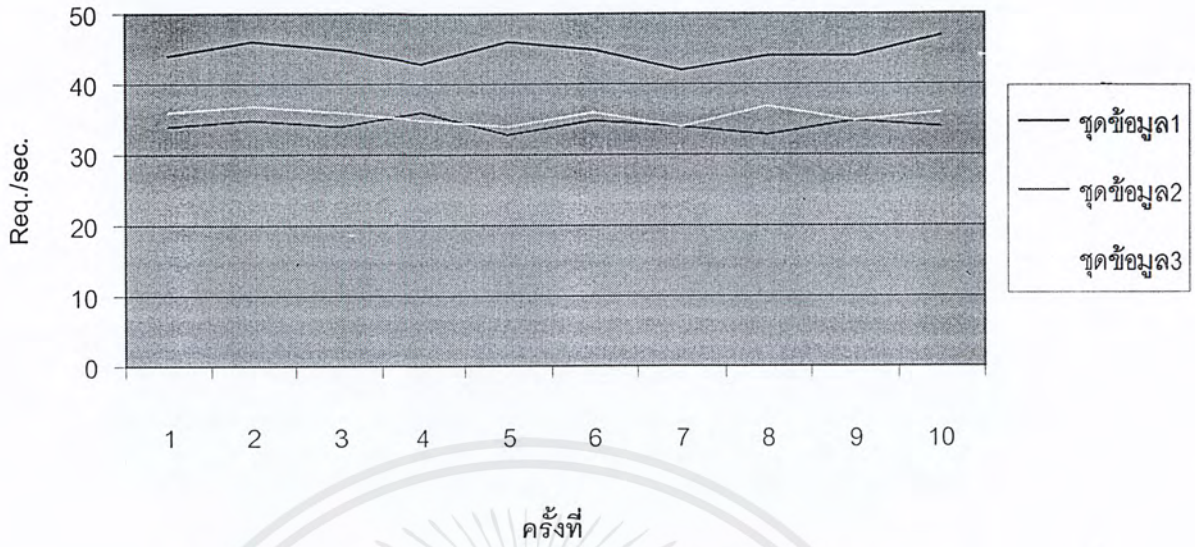
และเมื่อทำการทดสอบกับอัลกอริทึมอื่นๆก็ได้ค่าเฉลี่ยต่างๆดังตารางที่ 6.2

ชนิดของอัลกอริทึม	จำนวน Req./sec.	อัตรา Throughput (Kbyte/sec.)
Weighted Least Connection	44.6	243.3
Round Robin	34.3	187.1
Weighted Round Robin	35.6	193

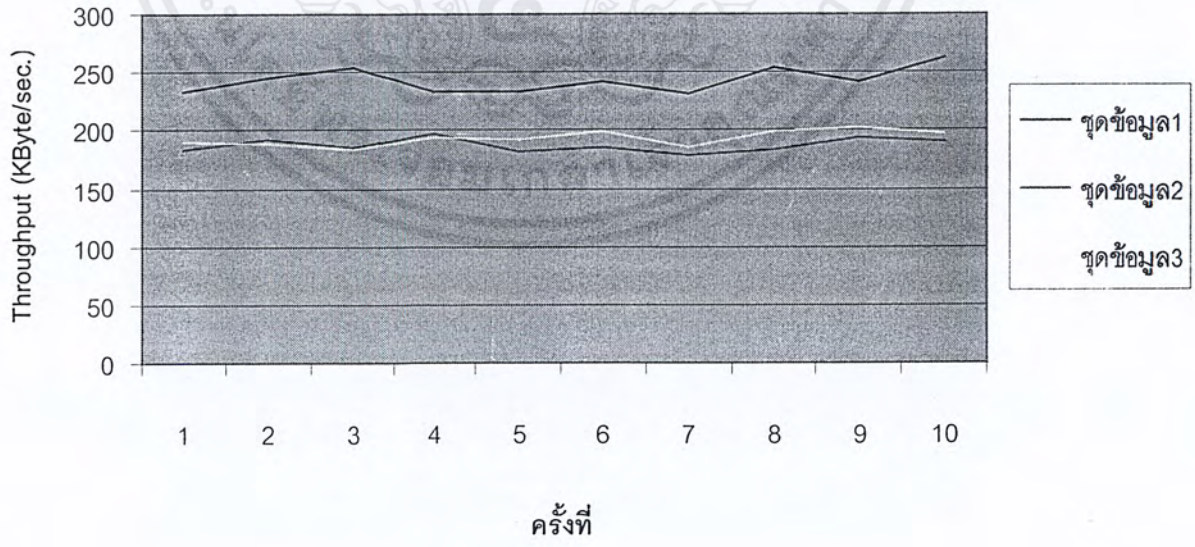
ตารางที่ 6.2 แสดงค่าเฉลี่ยของจำนวน Req./sec. และอัตรา Throughput ของแต่ละอัลกอริทึม

จากตารางจะเห็นได้ว่าอัลกอริทึมแบบ Weighted Least Connection จะให้ประสิทธิภาพที่ดีที่สุด เนื่องจากจะมีการตรวจสอบจำนวน Connection ของแต่ละ Real server และมีการให้น้ำหนักแก่แต่ละ Real server อีกด้วย

รูปที่ 6.3 และ 6.4 จะเปรียบเทียบให้เห็นถึงประสิทธิภาพของแต่ละอัลกอริทึม เมื่อมีจำนวน Real server เท่ากันคือ 3 เครื่อง



รูปที่ 6.3 แสดงการเปรียบเทียบจำนวน Req./sec. ของแต่ละอัลกอริทึม



รูปที่ 6.4 แสดงการเปรียบเทียบอัตรา Throughput ของแต่ละอัลกอริทึม

6.2 บทวิจารณ์

จากผลการทดสอบที่ผ่านๆมา จะเห็นได้ว่ายิ่งเพิ่มจำนวน Real server ให้มากขึ้น ประสิทธิภาพโดยรวมก็จะยิ่งเพิ่มขึ้น เนื่องจากมีเซิร์ฟเวอร์มาช่วยรองรับบริการมากขึ้น แต่ทั้งนี้ประสิทธิภาพจะดีมากเพียงใดนั้นก็จะขึ้นอยู่กับอัลกอริทึมที่เลือกใช้ด้วย โดยจะเห็นได้ว่าอัลกอริทึม Weighted Least Connection จะเหมาะกับการใช้งานมากที่สุด

อย่างไรก็ดีหาก Real server ที่ทำการเพิ่มเข้าไปนั้นมีประสิทธิภาพด้อยกว่าระบบเดิมมากๆ ก็อาจจะไม่ช่วยให้ระบบเดิมมีประสิทธิภาพที่ดีขึ้นแต่อย่างไร และบางทีอาจเป็นผลให้ประสิทธิภาพลดลงอีกด้วย เนื่องจากจะไปเพิ่ม traffic ภายในระบบ

ปัจจัยสำคัญอีกประการที่มีผลต่อประสิทธิภาพก็คือ ระบบเน็ตเวิร์กที่ใช้ เพราะจำเป็นที่จะต้องมีการสื่อสารกันระหว่างตัว Load balancer และ Real server แต่ละเครื่อง หากมีระบบเน็ตเวิร์กที่มีประสิทธิภาพสูงก็ย่อมทำให้ประสิทธิภาพการให้บริการดีขึ้น

จากการที่ได้ลงมือติดตั้งและทดสอบระบบ Web Server Clustering มาชั่วระยะเวลาหนึ่ง ทำให้รับรู้ได้ถึงประสิทธิภาพและการใช้งานได้จริงของระบบนี้ และประจักษ์ถึงความจริงที่ว่าแนวโน้มของการนำมาใช้แทนเครื่องเซิร์ฟเวอร์ที่ใช้กันอยู่จะต้องเพิ่มขึ้นอย่างแน่นอน

บรรณานุกรม

[1] W. Zhang and et al. Linux virtual server project.
http://www.LinuxVirtualServer.org/, 1998

[2] M. Wangsmo. White paper: Piranha - load-balanced web and ftp clusters.
http://www.redhat.com/support/wpapers/piranha/, 1999

[3] R. S.Engelschall. Load balancing your web site: Practical approaches for distributing http traffic. Web Technique Magazine, 315), May 1998.
http://www.webtechniques.com

[4] A. Roberson and et al. High-Availability linux project.
http://www.linux-ha.org/, 1998

[5] T. Brisco. Dns support for load balancing.
http://www.ietf.org/rfc/rfc1794.txt, April 1995. RFC 1794.