โปรแกรมสร้างค่าติดตั้งสำหรับอุปกรณ์เครือข่าย

Network Device Configuration Maker

นายาราษุทธ ตั้งกมลสถาพร
นางสาววรารัตน์ เชิดตระกูลชัย

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2546

โปรแกรมสร้างค่าติดตั้งสำหรับอุปกรณ์เครือข่าย

Network Device Configuration Maker

โดย

นายวรายุทธ ตั้งกมลสถาพร

นางสาววรารัตน์ เชิดตระกูลชัย

อาจารย์ที่ปรึกษา

อาจารย์ธนา หงษ์สุวรรณ

อาจารย์อัครเดช วัชระภูพงษ์

อาจารย์ธนัญชัย ตรีภาค

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2546

ปริญญานิพนธ์ ปีการศึกษา 2546

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง โปรแกรมสร้างค่าติดตั้งสำหรับอุปกรณ์เครือข่าย

Network Device Configuration Maker

คณะผู้จัดทำ นายวรายุทธ ตั้งกมลสถาพร รหัส 43010376

นางสาววรารัตน์ เชิดตระกูลชัย รหัส 43010377

..... อาจารย์ที่ปรึกษา

(อาจารย์ธนา หงษ์สุวรรณ)

..... อาจารย์ที่ปรึกษา

(อาจารย์อัครเดช วัชระภูพงษ์)

..... อาจารย์ที่ปรึกษา

(อาจารย์ธนัญชัย ตรีภาค)

โปรแกรมสร้างค่าติดตั้งสำหรับอุปกรณ์เครือข่าย

นายวรายุทธ ตั้งกมลสถาพร 43010376
นางสาววรารัตน์ เชิดตระกูลชัย 43010377
อาจารย์ธนา หงษ์สุวรรณ อาจารย์ที่ปรึกษา
อาจารย์อัครเดช วัชระภูพงษ์ อาจารย์ที่ปรึกษา
อาจารย์ธนัญชัย ตรีภาค อาจารย์ที่ปรึกษา
ปีการศึกษา 2546

บทคัดย่อ

ในปัจจุบัน มีการใช้งานเครือข่ายมากขึ้น หลายๆองค์กรมีเครือข่ายเป็นของตนเอง จึงทำให้งานประเภทการออกแบบเครือข่ายและการดูแลเครือข่ายมีมากขึ้นและมีความจำเป็น แต่การติดตั้งค่าอุปกรณ์เครือข่ายสำหรับ สวิตช์ และ เราเตอร์ ต้องอาศัยผู้ที่มีความรู้ความเชี่ยวชาญ เนื่องจากการตั้งค่าการทำงานต่างๆของสวิตช์และเราเตอร์ ต้องอาศัยการพิมพ์คำสั่งในลักษณะของ command line ผู้ใช้งานจะต้องจดจำการใช้คำสั่งต่างๆซึ่งมีเป็นจำนวนมาก ทำให้มีโอกาสพบความผิดพลาดสูง และในบางครั้งเป็นการกระทำที่ซ้ำซ้อน เมื่อต้องการตั้งค่าอุปกรณ์ สวิตช์ และ เราเตอร์ จำนวนมาก ทำให้เป็นการสิ้นเปลืองเวลาและค่าใช้จ่าย

ด้วยเหตุนี้ การพัฒนาเครื่องมือสำหรับช่วยเหลือผู้ดูแลระบบ จึงได้ถูกพัฒนาขึ้นเพื่อช่วยให้การออกแบบเครือข่ายมีความสะดวก ง่ายต่อการใช้งาน โดยใช้งานผ่าน user interface ที่สามารถแสดงการเชื่อมต่อของอุปกรณ์ และสามารถช่วยลดภาระในการติดตั้งและดูแลเครือข่าย ทำให้การติดตั้งระบบเป็นไปได้อย่างรวดเร็ว

Network Device Configuration Maker

Mr. Warayouth Thungkamolsataporn
Ms. Wararat Cherdtrakulchai
Mr. Thana Hongsuwan Advisor
Mr. Akkradach Watcharapupong Advisor
Mr. Thananchai Treepark Advisor

Abstract

Nowadays, network is more useful. Many organizations have the own network. This caused network design and network management is much more necessary. Since configuring switch and router may use command-line interface that should do by professional that make a higher expense otherwise it may cause error – bring in damage. Network Device Configuration Maker program focus on facility or ease to use, decreasing task and management to save time and money for managing network.

For this reason this project is established to support and help network administrator to decrease task and management by decreasing time and producing an ease from using program in graphic mode.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้จะไม่สามารถเสร็จสมบูรณ์ได้ถ้าไม่ได้รับความช่วยเหลือและการร่วมมือของบุคคลหลายๆ ฝ่ายด้วยกัน โดยเฉพาะอย่างยิ่งบุคคลผู้ซึ่งเป็นผู้จุดประกายความคิดให้เกิดหัวข้อปริญญานิพนธ์นี้ขึ้นมา นั่นก็คืออาจารย์ธนา หงษ์สุวรรณ อาจารย์อัครเดช วัชระภูพงษ์ และอาจารย์ ธนัญชัย ตรีภาค อาจารย์ที่คอยให้คำปรึกษาปริญญานิพนธ์ตลอดเวลา และขอบพระคุณคณาจารย์ทุกท่านในภาควิชาวิศวกรรมคอมพิวเตอร์ที่ได้ให้คำแนะนำ และความรู้ทางด้านคอมพิวเตอร์

ขอขอบคุณภาควิชาวิศวกรรมคอมพิวเตอร์โดยเฉพาะห้องวิจัยและพัฒนาการรักษาความปลอดภัยข้อมูล (ISAG) ที่ได้เอื้อเฟื้อสถานที่ ให้คณะผู้จัดทำได้ทำการวิจัย และช่วยอำนวยความสะดวกต่างๆขอขอบคุณเพื่อนๆ พี่ๆ น้องๆ ชาว ISAG ที่คอยให้ความช่วยเหลือในการทำงานตลอดเวลา เป็นที่ปรึกษายามมีปัญหา รวมทั้งให้ยืมทรัพยากรที่จำเป็นต่างๆ

ที่สำคัญและขาดมิได้ คือ ต้องขอขอบพระคุณบิดา มารดาที่ได้ให้กำเนิด คอยสั่งสอน และให้การสนับสนุนการศึกษา กิจกรรมต่างๆ นับเป็นพระคุณที่หาใดเปรียบมิได้ ทางคณะผู้จัดทำขออกราบขอบพระคุณมา ณ ที่นี้ด้วย

คณะผู้จัดทำ

สารบัญ

หน้าที่

สารบัญ(ต่อ)

หน้าที่

สารบัญ(ต่อ)

หน้าที่

สารบัญ(ต่อ)

หน้าที่

สารบัญ(ต่อ)

หน้าที่

สารบัญ(ต่อ)

หน้าที่

สารบัญภาพประกอบ

หน้าที่

สารบัญภาพประกอบ(ต่อ)

หน้าที่

สารบัญภาพประกอบ(ต่อ)

หน้าที่

สารบัญภาพประกอบ(ต่อ)

หน้าที่

สารบัญตาราง

หน้าที่

บทที่ 1
บทนำ

1.1 ความสำคัญและที่มา

เนื่องจากการตั้งค่าการทำงานต่างๆของสวิตช์และเราเตอร์ ต้องอาศัยการพิมพ์คำสั่งในลักษณะของ command line ผู้ใช้งานจะต้องจดจำการใช้คำสั่งต่างๆซึ่งมีเป็นจำนวนมาก ทำให้มีโอกาสพบความผิดพลาดสูง และในบางครั้งเป็นการกระทำที่ซ้ำซ้อน เมื่อต้องการตั้งค่าอุปกรณ์ สวิตช์ และ เราเตอร์ จำนวนมาก ทำให้เป็นการสิ้นเปลืองเวลาและค่าใช้จ่าย

ด้วยเหตุนี้ การพัฒนาเครื่องมือสำหรับช่วยเหลือผู้ดูแลระบบ จึงได้ถูกพัฒนาขึ้นเพื่อช่วยให้การออกแบบเครือข่ายมีความสะดวก ง่ายต่อการใช้งาน โดยใช้งานผ่าน user interface ที่สามารถแสดงการเชื่อมต่อของอุปกรณ์ และสามารถช่วยลดภาระในการติดตั้งและดูแลเครือข่าย ทำให้การติดตั้งระบบเป็นไปได้อย่างรวดเร็ว

1.2 วัตถุประสงค์ของปริญญานิพนธ์

ปริญญานิพนธ์นี้จัดทำขึ้นเพื่อสร้างโปรแกรมสร้างค่าติดตั้งอุปกรณ์เครือข่ายสวิตช์และเราเตอร์ผ่าน graphic user interface ที่ใช้งานง่าย มีความสามารถในการตั้งค่าพื้นฐานที่จำเป็นและถูกใช้งานบ่อยครั้ง และสามารถนำค่าติดตั้งที่ได้จากโปรแกรมมาตั้งค่าอุปกรณ์สวิตช์และเราเตอร์ได้จริง

1.3 ขอบเขตของปริญญานิพนธ์

(1) เป็นโปรแกรมที่ใช้ในการตั้งค่าอุปกรณ์ซิสโก้สวิตช์รุ่น 2950 ซีรี่ส์ และ ซิสโก้เราเตอร์รุ่น 1760 และ 2620 ซีรี่ส์

(2) สามารถตั้งค่าทั่วไป เช่น ชื่ออุปกรณ์, ไอพีแอดเดรสของอุปกรณ์สวิตช์และเราเตอร์ได้

(3) สามารถกำหนดโหมดการทำงานของอินเทอร์เฟสให้กับแต่ละอินเทอร์เฟสของสวิตช์ได้

(4) สามารถสร้างและกำหนดค่าให้กับอินเทอร์เฟสที่ใช้ในการจัดการได้ ได้แก่ วีแลนอินเทอร์เฟส และ พอร์ตแชนแนลอินเทอร์เฟส

(5) สามารถกำหนดไอพีแอดเดรสให้กับแต่ละอินเทอร์เฟสของเราเตอร์

(6) สามารถกำหนดโปรโตคอลเลือกเส้นทางให้แก่เราเตอร์ได้

(7) สามารถกำหนดแอคเซสลิสต์เพื่อใช้ในการอนุญาต และป้องกันแพ็กเก็ตที่เข้าและออกจากอุปกรณ์สวิตช์ เราเตอร์

(8) สามารถกำหนดการเข้าใช้งานเพื่อ เปลี่ยนแปลงแก้ไขอุปกรณ์สวิตช์ เราเตอร์ ผ่านคอนโซลพอร์ตและเครือข่าย

1.4 ขั้นตอนการดำเนินงาน

(1) ศึกษาและรวบรวมคำสั่งที่ใช้ในการตั้งค่าอุปกรณ์สวิตช์และเราเตอร์

(2) ออกแบบคลาส และกำหนดรายละเอียดต่างๆ โดยแบ่งออกเป็นส่วนของสวิตช์ และเราเตอร์

(3) วิเคราะห์ค่าติดตั้งที่ได้จากการใช้คำสั่งในการกำหนดค่าให้กับสวิตช์และเราเตอร์

(4) ออกแบบโปรแกรมในส่วนที่ติดต่อกับผู้ใช้งาน

(5) เขียนโปรแกรมเพื่อกำหนดรายละเอียดการทำงาน

(6) ทดสอบและตรวจหาข้อผิดพลาด

(7) ทำเอกสารคู่มือการใช้งาน

บทที่ 2
ความรู้พื้นฐาน

2.1 เครือข่ายคอมพิวเตอร์ (Computer Network)

เครือข่ายคอมพิวเตอร์ หมายถึง เครื่องคอมพิวเตอร์ตั้งแต่สองเครื่องขึ้นไปที่เป็นอิสระต่อกัน นำมาเชื่อมต่อถึงกันได้โดยไม่คำนึงถึงระยะทางระหว่างเครื่องทั้งสอง โดยมีจุดประสงค์เพื่อให้เครื่องคอมพิวเตอร์สามารถติดต่อสื่อสาร และแลกเปลี่ยนข้อมูลระหว่างกันได้ เครือข่ายคอมพิวเตอร์เริ่มจากเครือข่ายขนาดเล็กภายในองค์กรที่เชื่อมโยงกันภายใต้สภาพพื้นที่จำกัดซึ่งเรียกว่า *เครือข่ายท้องถิ่น* (LAN: Local Area Network) เมื่อเชื่อมโยงเครือข่ายเข้าด้วยกัน และขยายขอบเขตครอบคลุมพื้นที่ระหว่างเมือง หรือระหว่างประเทศ ก็จะเรียกเครือข่ายนั้นว่า *เครือข่ายวงกว้าง* (WAN: Wide Area Network)

2.2 โปรแกรมสำหรับเครือข่าย (Network Software)

2.2.1 ลำดับขั้นของโพรโตคอล (Protocol Hierarchies)

เพื่อเป็นการลดความซับซ้อนของการออกแบบโปรแกรมทั้งระบบในคราวเดียวกัน ระบบโปรแกรมเครือข่ายส่วนมากจะแบ่งเป็นการทำงานออกเป็นหลายระดับ หรือหลายชั้น แต่ละชั้นจะสร้างฟังก์ชันการทำงานขึ้น โดยอาศัยการทำงานของฟังก์ชันต่าง ๆ ที่สร้างไว้ในชั้นระดับล่างลงมา จำนวนชั้น, ชื่อที่เรียก และฟังก์ชันการทำงานของเครือข่ายต่าง ๆ จะแตกต่างกันออกไป อย่างไรก็ตาม ทุกระบบจะมีแนวความคิดอย่างเดียวกัน คือการเรียกใช้บริการจากชั้นล่าง และการให้บริการแก่ชั้นบน โดยซ่อนรายละเอียดและ ความซับซ้อนของฟังก์ชันในแต่ละชั้นไว้ภายใน

ในการสื่อสารที่เกิดขึ้นระหว่างผู้ส่งข้อมูลกับผู้รับข้อมูลนั้น จะเสมือนว่าเป็นการติดต่อในชั้นเดียวกัน โดยในแต่ละชั้นจะมีโพรโตคอลของตนเอง ซึ่งจะช่วยให้ผู้ส่งข้อมูลและผู้รับข้อมูลสามารถติดต่อกันได้ แต่ในความจริงแล้ว การสื่อสารจะเกิดขึ้นจริงโดยผ่านสายสื่อสารที่อยู่ชั้นหนึ่งเท่านั้น ข้อมูลที่สื่อสารในระหว่างชั้นต่าง ๆ จะถูกส่งต่อกันเป็นลำดับจากชั้นบนสู่ชั้นล่าง จากฝ่ายของทางผู้ส่งข้อมูล และทางฝ่ายของผู้รับข้อมูลจะทำการรับข้อมูลจากชั้นหนึ่ง แล้งจึงถูกแยกข้อมูลออกไปทีละชั้น จนถึงชั้นบนสุดเพื่อใช้ในการประมวลผลต่อไป

2.2.2 ระบบมาตรฐาน OSI (The OSI Reference Model)

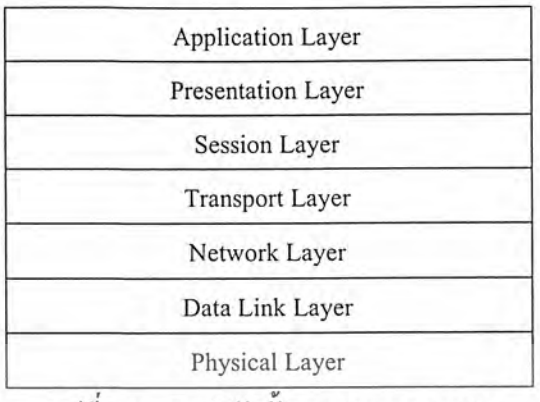
เมื่อคอมพิวเตอร์ของเรามีการรับส่งข้อมูลกับคอมพิวเตอร์เครื่องอื่น ๆ การเชื่อมต่อคอมพิวเตอร์หลาย ๆ เครื่องเข้าด้วยกันเป็นระบบเครือข่ายก็เกิดขึ้น อย่างไรก็ตาม การเชื่อมต่อคอมพิวเตอร์คนละระบบหรือคนละยี่ห้อเป็นสิ่งที่ทำได้ยากในยุคแรก ๆ ของการสื่อสารข้อมูล เนื่องจากขาดมาตรฐานส่วนกลางที่จำเป็นต้องใช้ในการรับส่งข้อมูล ส่วนมากแต่ละยี่ห้อก็จะมีมาตรฐานของตนเอง ซึ่งเข้ากันไม่ได้กับของยี่ห้ออื่น ทำให้ผู้ใช้ต้องผูกติดอยู่กับผู้ผลิตแต่ละยี่ห้อ และเป็นขีดจำกัดในการเชื่อมต่อคอมพิวเตอร์คนละชนิด ไม่ให้รับส่งข้อมูลกันได้ ระบบคอมพิวเตอร์ในยุคนั้น จึงเป็นระบบปิด (Closed System) นั่นเอง

ปัญหานี้ ทำให้หน่วยงานกำหนดมาตรฐานสากล คือ International Standards Organization หรือ ISO จัดการกำหนดโครงสร้างทั้งหมดที่จำเป็นต้องใช้ในการสื่อสารข้อมูลจากคอมพิวเตอร์ระบบหนึ่งไปยังคอมพิวเตอร์อีกระบบหนึ่งขึ้น จุดมุ่งหมายก็เพื่อเปิดช่องทางให้ข้อมูลที่เก็บอยู่ในระบบคอมพิวเตอร์หนึ่ง ๆ รับส่งไปยังคอมพิวเตอร์ที่เป็นระบบเดียวกัน หรือต่างระบบได้อย่างอิสระ โดยไม่ขึ้นกับผู้ผลิตอย่างที่เป็นอยู่ในอดีต ซึ่งเป็นการทำงานแบบที่เรียกว่า ระบบเปิด (Open systems) เราเรียกโครงสร้างของมาตรฐานการรับส่งข้อมูลนี้ว่า (Open System Interconnection) หรือ OSI ซึ่งจัดทำขึ้นราวกลางปี ค.ศ.1970 โดยองค์การ International Standards Organization (ISO) และใช้อ้างอิงกันมาจนถึงในยุคปัจจุบัน

โมเดล OSI มีทั้ง 7 ชั้นด้วยกัน หรือที่เราเรียกว่า OSI 7-Layers Model โดยตัวโครงสร้างจะเน้นความสำคัญของรูปแบบการติดต่อสื่อสาร ระหว่างระบบเปิด (open systems) กับระบบเปิด จึงสามารถนำไปใช้อ้างอิงได้ในระดับสากลอย่างแท้จริง

แนวความคิดของการกำหนดมาตรฐานเป็นแบบชั้นสื่อสาร (layers) คือ

1. ชั้นสื่อสารแต่ละชั้นถูกกำหนดขึ้นมาตามบทบาทที่แตกต่างกัน
2. แต่ละฟังก์ชันในชั้นใด ๆ จะต้องกำหนดขึ้นมาโดยใช้แนวความคิดในระดับสากลเป็นวัตถุประสงค์หลัก
3. ขอบเขตความรับผิดชอบของแต่ละชั้น จะต้องกำหนดขึ้นมา เพื่อจำกัดปริมาณการแลกเปลี่ยนข้อมูลและ ผลกระทบข้างเคียงระหว่างการติดต่อให้มีน้อยที่สุด
4. ในแต่ละชั้น จะเสมือนเชื่อมต่ออยู่กับชั้นเดียวกันของอีกฝั่งหนึ่ง โดยทำงานเสมือนกับว่า มีการติดต่อรับส่งข้อมูลกับกลไกในชั้นเดียวกัน แต่จะมีเพียงชั้นหนึ่ง ซึ่งเป็นชั้นล่างสุดเท่านั้นที่มีการเชื่อมต่อ และรับส่งข้อมูลผ่านสายส่งข้อมูลระหว่างคอมพิวเตอร์ทั้งสองระบบ
5. ในแต่ละชั้น ที่ทำหน้าที่รับส่งข้อมูลจะมีการติดต่อรับส่งข้อมูลกับชั้นที่อยู่ติดกับตนเองเท่านั้น ไม่สามารถติดต่อรับส่งข้อมูลข้ามชั้นได้



รูปที่ 2-1 แสดงลำดับชั้นของมาตรฐาน OSI

2.2.2.1 ชั้นสื่อสารกายภาพ (The Physical Layer)

ชั้นนี้ คือชั้นที่ 1 หรือชั้นล่างสุด จะทำงานเกี่ยวข้องโดยตรงกับอุปกรณ์สื่อสารต่าง ๆ ทำหน้าที่ในการกำหนดวิธีการควบคุมการรับ และ การส่งข้อมูลระหว่างเครื่องคอมพิวเตอร์ในระดับบิต ได้แก่ การส่งบิต 0 และ 1 จะแทนด้วยกระแสไฟฟ้ากี่โวลต์ แต่ละบิตในระยะเวลาในการส่งนานเท่าไหร่ การส่งเป็นแบบทางเดียวหรือสองทาง เป็นต้น จะเห็นได้ว่า การทำงานในชั้นนี้จะเกี่ยวข้องกับการทำงานของอุปกรณ์สัญญาณไฟฟ้า (หรือสัญญาณใด ๆ) ขั้นตอนในการใช้อุปกรณ์เหล่านั้น และความสัมพันธ์กับสื่อที่ใช้รับ – ส่งสัญญาณ

2.2.2.2 ชั้นสื่อสารการเชื่อมต่อข้อมูล (The Data Link Layer)

หน้าที่หลักของชั้นนี้คือ ทำการรวบรวมข้อมูลจากชั้นที่ 1 ตรวจสอบความถูกต้องของข้อมูล แล้วส่งข้อมูลที่ปราศจากข้อผิดพลาดนี้ให้กับชั้นที่ 3 ต่อไป โดยปกติผู้ส่งข้อมูลจะแบ่งข้อมูลที่มีความยาวมากออกเป็นกลุ่มข้อมูลย่อย ๆ แต่ละส่วนย่อยเรียกว่า ดาต้าเฟรม (Data Frame) ซึ่งจะมีขนาดคงที่ ชุดของดาต้าเฟรมสำหรับข้อมูลที่ต้องการส่งไปให้ผู้รับก็จะถูกส่งไปทีละเฟรม ตั้งแต่เฟรมแรกไปจนครบทุกเฟรม ข้างฝ่ายผู้รับจะตอบสนองโดยการส่งดาต้าเฟรมพิเศษ เรียกว่า เฟรมตอบรับ (Acknowledgement frame) ไปถึงผู้ส่งเพื่อเป็นการบอกให้ทราบว่า ได้รับข้อมูลแล้ว กระบวนการรับ-ส่งข้อมูลนี้ก็จะเสร็จสิ้นสมบูรณ์

การรับ-ส่งข้อมูลในชั้นที่ 1 นั้นจะไม่รับรู้ในเรื่องโครงสร้างข้อมูล คือจะมองเห็นข้อมูลว่าเป็นบิต 0 หรือบิต 1 กลุ่มหรือชุดหนึ่งที่เรียงตามลำดับ เรียกว่า กระแสบิต (Bit Stream) จึงเป็นหน้าที่ของชั้นที่ 2 ในการทำการตรวจสอบความถูกต้อง ซึ่งทำได้โดยการเพิ่มข้อมูลสำหรับการตรวจสอบติดไว้กับข้อมูลทุกเฟรม

การส่งข้อมูลผ่านระบบเครือข่ายใด ๆ ก็ตาม ข้อมูลที่ส่งนั้นมีโอกาสที่จะเสียหายหรือสูญหายไปเลยก็ได้ โปรแกรมในชั้นที่ 2 จะต้องสามารถตรวจสอบความผิดปกตินี้ได้เอง หรืออาจตอบสนองต่อการตรวจพบโดยโปรแกรมในชั้นที่ 1 เมื่อพบความผิดปกตินี้แล้วก็จะต้องมีวิธีในการแก้ไข เช่น แจ้งให้ผู้ส่งข้อมูลได้ส่งข้อมูลชุดเดิมกลับมาใหม่ (เรียกเฟรมนี้ว่า Duplicate Frame) อย่างไรก็ตาม การส่งข้อมูลซ้ำทำให้เกิดปัญหาตามมา

ในกรณีที่ชุดข้อมูลไม่ได้สูญหายไปไหน เพียงแต่ใช้เวลาเดินทางมากกว่าปกติ ดังนั้นข้อมูลชุดเดียวกันก็จะมาถึงผู้ใช้ทั้งสองเฟรม โปรแกรมในชั้นนี้จึงต้องหาวิธีการตรวจสอบและกำจัดเฟรมที่ซ้ำออกไป

2.2.2.3 ชั้นสื่อสารเครือข่าย (The Network Layer)

ชั้นนี้มีหน้าที่รับผิดชอบในการควบคุมการติดต่อรับ-ส่งข้อมูลระหว่างเครื่องคอมพิวเตอร์ (โหนด) ต่าง ๆ ในระบบเครือข่ายให้เป็นไปด้วยความเรียบร้อย สิ่งที่สำคัญที่สุด คือการกำหนดเส้นทางเดินของข้อมูลจากโหนดผู้ส่งข้อมูลไปตามโหนดต่าง ๆ จนถึงโหนดผู้รับข้อมูลในที่สุด โฮสต์บางกลุ่มจะกำหนดเส้นทางเดินข้อมูลโดยศึกษาระบบเครือข่ายแล้วสร้างตารางเส้นทางเดินข้อมูลแบบถาวร โฮสต์บางกลุ่มจะกำหนดเส้นทางเดินข้อมูลในตอนเริ่มต้นของการสื่อสาร ดังนั้นการสื่อสารในครั้งต่อไป (ติดต่อกับโหนดเดิม) อาจจะเปลี่ยนไปใช้เส้นทางอื่นได้ โฮสต์ในกลุ่มที่มีวิธีการซับซ้อนมาก จะกำหนดเส้นทางเดินข้อมูลในระดับแพ็กเก็ต กรณีที่มีผู้ส่งข้อมูลพร้อม ๆ กันหลาย ๆ จุดจะทำให้เกิดความคับคั่งของข้อมูลคล้ายกับสภาวะจราจรในชั่วโมงเร่งด่วน ซึ่งมีปริมาณรถยนต์มากจนทำให้การจราจรติดขัด โฮสต์ในกลุ่มนี้ก็จะปรับเส้นทางเดินข้อมูลของแต่ละแพ็กเก็ตให้เหมาะสมกับสภาวะของระบบเครือข่ายอยู่ตลอดเวลา

การส่งผ่านข้อมูลในระบบเครือข่ายอาจมีการบันทึก ผู้ส่ง ผู้รับ และปริมาณข้อมูลที่ไหลผ่านโฮสต์หรือ เราเตอร์ต่าง ๆ เพื่อประโยชน์ทางด้านการคิดค่าบริการซึ่งจะมีความซับซ้อนมากขึ้นถ้าข้อมูลไหลผ่านระบบเครือข่ายย่อยที่มีการคิดอัตราค่าบริการต่างกัน

2.2.2.4 ชั้นจัดการนำส่งข้อมูล (The Transport Layer)

โปรแกรมในชั้นนี้ มีหน้าที่หลักในการรับข้อมูลมาจากชั้นที่ 5 ซึ่งอาจต้องแบ่งข้อมูลออกเป็นแพ็กเก็ตขนาดย่อม (ในกรณีที่ข้อมูลมีปริมาณมาก) หลาย ๆ แพ็กเก็ต แล้วจึงส่งข้อมูลทั้งชุดต่อไปให้โปรแกรมในชั้นที่ 3 ทางด้านโปรแกรมในชั้นที่ 4 ก็จะทำหน้าที่ประกอบแพ็กเก็ตชุดนี้ให้กลับมารวมกันเป็นข้อมูลชุดเดิม

ในภาวะปกติ การเชื่อมต่อการสื่อสารจะเป็นการจัดตั้งหน้าต่างสื่อสาร (Session) ระหว่างผู้ส่งและผู้รับตามที่เกิดขึ้น ถ้าต้องการเพิ่มประสิทธิภาพก็อาจสร้างโพสเซสของโปรแกรมในชั้นนี้ขึ้นมาหลาย ๆ โพรเซสเพื่อช่วยกันจัดส่งข้อมูลให้เร็วขึ้น แต่ถ้าเน้นในด้านความประหยัดก็อาจทำในทางตรงกันข้ามนั่นคือ การยุบรวมโพรเซสหลาย ๆ โพรเซสให้เหลือจำนวนน้อยลงแล้วจึงจัดการให้โพรเซสที่เหลืออยู่ทำการส่งข้อมูลทั้งหมดโดยการใช้ช่องสื่อสารร่วมกัน

โปรแกรมในชั้นนี้เป็นผู้กำหนดประเภทของการให้บริการต่าง ๆ รวมไปถึงการอำนวยความสะดวกในการใช้ระบบเครือข่ายซึ่งแบ่งออกได้เป็น 3 ประเภท ประเภทแรกเป็นการให้บริการแบบจุด-ต่อ-จุด โดยเน้นการรับประกันความถูกต้องของข้อมูลเป็นสำคัญ ประเภทที่สองเน้นการให้บริการข้อมูล ข้อมูลในระดับแพ็กเก็ตซึ่งแม้ว่าจะไม่รับประกันการสูญหายของข้อมูลแต่ก็ให้ความคล่องตัวสูงกว่าแบบแรก (การรับประกันความ

ถูกต้องของข้อมูลสามารถทำในชั้นอื่นได้) ประเภทที่สามเป็นการส่งข้อมูลแบบกระจายข่าวเพื่อประโยชน์ในการส่งข้อมูลชุดเดียวกันไปยังผู้ใช้หลายจุดพร้อมกัน

โปรแกรมในชั้นนี้ติดต่อถึงกันผ่านช่องสัญญาณเสมือน (Virtual Channel) ระหว่างผู้ส่งและผู้รับโดยตรงเรียกว่า เป็นการติดต่อกับ end-to-end connection ในขณะที่โปรแกรมในสามชั้นแรกนั้นเป็นการติดต่อแบบจุด-ต่อ-จุด ซึ่งผู้รับอาจไม่ใช่ผู้รับข้อมูลแต่เป็นเพียงโหนดตัวกลางในการรับแล้วส่งข้อมูลต่อไปตามเส้นทางเดินข้อมูลที่ถูกกำหนดไว้

นอกจากการใช้ช่องสื่อสารร่วมกันแล้วโปรแกรมในชั้นนี้ จะต้องมีความสามารถในการจัดตั้งหน้าต่างสื่อสารกับโหนดอื่น ๆ ในระบบเครือข่ายและจัดการยกเลิกเมื่อการสื่อสารสิ้นสุดลง โปรแกรมในชั้นนี้ยังต้องมีวิธีการกำหนดการตั้งชื่อให้แก่ตนเองและแนะนำให้ผู้อื่นในระบบได้รู้จัก รวมทั้งการควบคุมการไหลของข้อมูล ซึ่งมีทั้งในระดับโฮสต์และระดับเราเตอร์โดยมีวัตถุประสงค์ในการควบคุมการรับและส่งข้อมูลโดยเฉพาะในกรณีที่ผู้ส่งจัดการส่งข้อมูลเร็วเกินกว่าทางผู้ใช้จะทำงานได้ทัน

2.2.2.5 ชั้นหน้าต่างสื่อสาร (The Session Layer)

ชั้นนี้ ทำหน้าที่เป็นผู้กำหนดวิธีการควบคุมการเชื่อมต่อระหว่างผู้รับข้อมูลและผู้ส่งข้อมูลตั้งแต่เริ่มต้นการสื่อสารไปจนยุติการสื่อสาร เช่น การติดต่อขอใช้โฮสต์จากเครื่องคอมพิวเตอร์ที่อยู่ไกลออกไป (Remote Login) โดยภาพรวมแล้ว การให้บริการในชั้นนี้จะคล้ายกับบริการที่มีอยู่ในชั้นที่ 4 แต่ในชั้นนี้จะให้บริการหลายอย่างที่เป็นประโยชน์มากกว่าสำหรับการประยุกต์ใช้งานบางประเภท

หน้าที่สำคัญอย่างหนึ่งคือ บริหารการแลกเปลี่ยนข่าวสาร อันได้แก่การกำหนดให้การแลกเปลี่ยนข่าวสารเป็นไปแบบสองทางในเวลาเดียวกัน (Full duplex) หรือถ้าเป็นการสื่อสารแบบทางเดียวแต่สลับทิศได้ (Half duplex) ก็จะต้องเป็นผู้จัดลำดับให้ทั้งผู้ใช้และผู้ส่งทำการส่งข้อมูลได้คล้ายกับการควบคุมสับหลีกรถไฟ

สำหรับการสื่อสารประเภทที่ต้องใช้โทเก้น (Token) โปรแกรมในชั้นนี้ จะเป็นผู้บริหารการใช้โทเก้นเพื่อให้โหนดต่าง ๆ ในระบบนี้ผลัดเปลี่ยนการครอบครองโทเก้นอย่างเป็นธรรม หรือถูกต้องตามลำดับความสำคัญ

หน้าที่อีกประการหนึ่งได้แก่การแก้ปัญหาความล้มเหลวในการส่งข้อมูลขนาดใหญ่มากระหว่างโหนดต่าง ๆ ในกรณีที่การส่งข้อมูลเกิดล้มเหลวกลางคันโดยไม่มีการแก้ไขใด ๆ โหนดทั้งสองก็จะต้องเริ่มต้นใหม่หมด ถ้าเกิดการล้มเหลวขึ้นอีกก็จะต้องเริ่มต้นใหม่อีก วิธีการแก้ไขวิธีหนึ่งก็คือการแทรกจุดตรวจสอบความถูกต้อง (Checkpoints) เข้าไปจำนวนหนึ่ง โดยจำนวนขึ้นกับปริมาณข้อมูล ในระหว่างการส่งข้อมูล จุดตรวจสอบทั้งหมดจะต้องถูกแทรกเข้าไปในข้อมูล ที่ตำแหน่งเดียวกันของทั้งผู้ส่งและผู้รับซึ่งเรียกว่าการ Synchronization หากเกิดการล้มเหลวขึ้น โปรแกรมในชั้นนี้ของผู้รับก็จะค้นหาจุดตรวจสอบจุดสุดท้ายก่อนการล้มเหลวเพื่อลบข้อมูลส่วนที่อยู่หลังจุดตรวจสอบนั้นทิ้งไป แล้วแจ้งให้ผู้ส่งเริ่มต้นการส่งข้อมูลใหม่จากจุดตรวจสอบนั้นแทนที่จะต้องเริ่มต้นใหม่ทั้งหมด

2.2.2.6 ชั้นนำเสนอข้อมูล (The Presentation Layer)

โปรแกรมที่ทำงานในระดับชั้นต้น ๆ นั้นที่ได้กล่าวมาแล้วนั้น จะให้ความสนใจในประสิทธิภาพของการรับ-ส่งข้อมูลและมองเห็นว่าข้อมูลคือกระแสบิต (Bit Stream) หรือกระแสไบต์ (Byte Stream) เท่านั้น โปรแกรมในชั้นนี้จะมองข้อมูลว่าเป็นสิ่งที่มีรูปแบบ (Syntax) และความหมาย (Semantics) มากกว่ากระแสบิตหรือไบต์ ความแตกต่างของการให้ความหมายข้อมูลของเครื่องคอมพิวเตอร์ในระบบต่าง ๆ เป็นปัญหาที่จะต้องได้รับการแก้ไขในระดับส่วนรวมไม่ใช่ให้แต่ละฝ่ายแก้ปัญหาโดยลำพัง การควบคุมรูปแบบและความหมายของข้อมูล การใช้รหัสแทนข้อมูล หรือการแทนข้อมูลด้วยระบบต่าง ๆ รวมไปถึงการเข้ารหัสและถอดรหัส สิ่งต่าง ๆ ที่กล่าวมานี้ล้วนแต่เป็นความรับผิดชอบของโปรแกรมในชั้นนี้

2.2.2.7 ชั้นสื่อสารการประยุกต์ (The Application Layer)

ในปัจจุบัน มีจอภาพเทอร์มินัล (Terminals) อยู่หลายร้อยชนิดทั่วโลกซึ่งส่วนใหญ่จะไม่สามารถใช้ทดแทนหรือใช้งานร่วมกันได้ การติดต่อระหว่างเครื่องคอมพิวเตอร์ที่อยู่คนละระบบเครือข่ายย่อยจึงไม่อาจสื่อสารกันได้โดยสมบูรณ์ โปรแกรมในชั้นประยุกต์จึงเข้ามามีบทบาทสำคัญสองด้านคือ การเป็นตัวกลาง หรือส่วนติดต่อระหว่างโปรแกรมประยุกต์ (Application Programs) กับโปรแกรมใน 6 ชั้นที่เหลือและ การกำหนดแบบมาตรฐานของจอ (Terminal Type)

การกำหนดแบบมาตรฐานของจอนั้น ไม่ได้เป็นการกำหนดวิธีสร้างจอเทอร์มินัลให้เหมือนกัน แต่จะคล้ายกับการสร้างจอเทอร์มินัลเสมือน (Virtual Terminal) ขึ้นบนจอเทอร์มินัลจริง ทั้งนี้เพื่อทำให้จอเทอร์มินัลทุกชนิดในโลกมีความเข้าใจตรงกัน เช่น ขนาดบริเวณที่ในการการแสดงผล การเคลื่อนย้ายตำแหน่งเคอร์เซอร์และการแสดงตัวอักษร ณ ตำแหน่งต่าง ๆ บนจอภาพ เป็นต้น จึงทำให้การใช้จอเทอร์มินัลเพื่อการสื่อสารบนระบบเครือข่ายเกิดขึ้นได้ แม้ว่าจะใช้จอเทอร์มินัลต่างแบบกันก็ตาม

บทที่ 3
ไอพีแอดเดรส

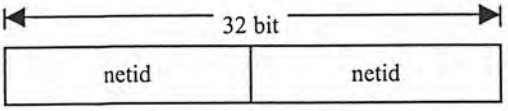
ไอพีแอดเดรสเป็นส่วนประกอบสำคัญอย่างหนึ่งในเครือข่ายทีซีพี/ไอพี ไอพีแอดเดรสเป็นแอดเดรสทางซอฟต์แวร์ประจำสถานีเครือข่ายผู้ออกแบบเครือข่ายจำเป็นต้องศึกษาและเข้าใจรูปแบบของไอพีแอดเดรสโดยละเอียด การจัดแบ่งแอดเดรสออกเป็นคลาส และการจัดแบ่งเครือข่ายย่อยหรือซับเน็ต รวมทั้งสามารถออกแบบซับเน็ตโดยเลือกใช้ไอพีแอดเดรสได้อย่างถูกต้อง เนื้อหาซึ่งจะกล่าวถึงในบทนี้มีดังนี้

3.1 ไอพีแอดเดรส

อุปกรณ์ที่เชื่อมเข้าเครือข่ายและสามารถทำงานตามข้อกำหนดของทีซีพี/ไอพีจะต้องมีแอดเดรสประจำอุปกรณ์นั้น อุปกรณ์ดังกล่าวอาจเป็นโฮสต์ เราเตอร์ เครื่องพิมพ์ หรือแม้กระทั่งอุปกรณ์สำนักงาน เช่นโทรศัพท์หรือเครื่องถ่ายเอกสาร ไอพีรุ่นสี่กำหนดให้ใช้ไอพีแอดเดรสขนาด 32 บิต อุปกรณ์ที่เชื่อมกับอินเทอร์เน็ตจะมีไอพีแอดเดรส 32 บิตประจำอินเทอร์เฟสที่ไม่ซ้ำกัน อุปกรณ์อย่างเราเตอร์จะมีหลายอินเทอร์เฟสซึ่งแต่ละอินเทอร์เฟสจะมีไอพีแอดเดรสหลายค่าตามจำนวนอินเทอร์เฟสโดยไม่ซ้ำค่ากัน แต่ถ้าเป็นเครื่องคอมพิวเตอร์หรือโฮสต์ปกติจะมีแค่อินเทอร์เฟสเดียว จึงมักเรียกว่าไอพีแอดเดรสเป็นแอดเดรสประจำโฮสต์

แอดเดรสขนาด 32 บิตมีจำนวนแอดเดรสรวมเท่ากับ 2^{32} (4,294,967,296) แต่เมื่อนำมาจัดสรรแล้วไม่สามารถใช้งานได้ครบทั้งหมด ไอพีแอดเดรสนิยมเขียนในรูปเลขฐานสิบ โดยแบ่งเลข 32 บิตเป็น 4 ไบต์ แต่ละไบต์แทนด้วยตัวเลขฐานสิบหนึ่งตัวคั่นแต่ละไบต์ใช้ด้วยเครื่องหมายจุด เช่น แอดเดรส 1001110 01101100 00000010 00000001 จะเขียนได้เป็น 161.246.2.1

แอดเดรสขนาด 32 บิต ประกอบขึ้นจากหมายเลขสองส่วนคือ เลขเครือข่าย (Network Number หรือ Network Identifier หรือ NetID) และ เลขโฮสต์ (Host Number หรือ Host Identifier หรือ HostID) เลขเครือข่ายใช้สำหรับจัดคลาสเครือข่าย ส่วนเลขโฮสต์ใช้ระบุหมายเลขโฮสต์ (หรืออีกนัยหนึ่งคืออินเทอร์เฟสของโฮสต์) ในเครือข่าย ไอพีแอดเดรสจึงแบ่งได้เป็นสองส่วนตามรูปที่ 4-1 จำนวนบิตที่ใช้สำหรับเลขเครือข่ายและเลขโฮสต์ขึ้นอยู่กับคลาสที่สังกัด



รูปที่ 3-1 รูปแบบของไอพีแอดเดรส

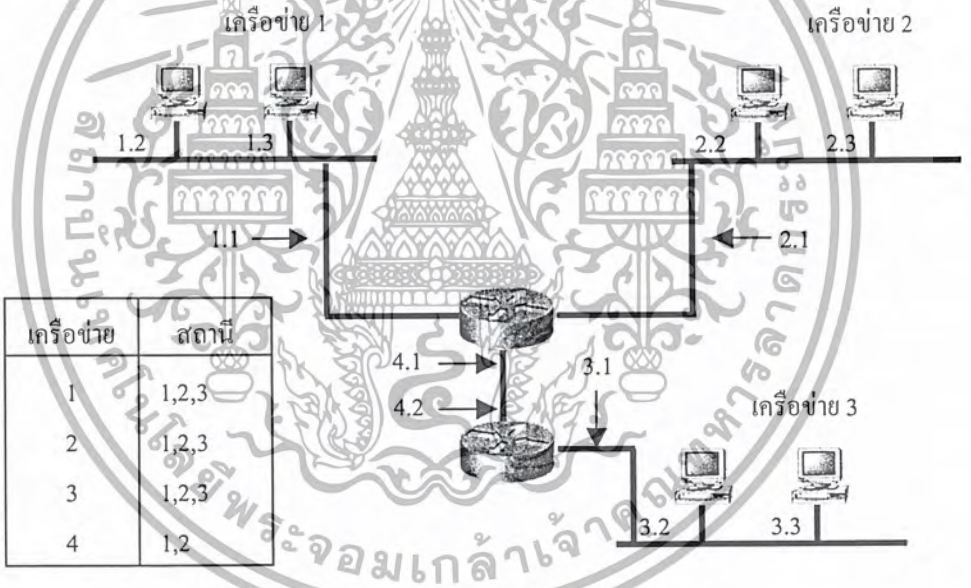
ในปัจจุบันฟิลด์กำหนดเลขเครือข่ายนิยมเรียกว่า พรีฟิกซ์เครือข่าย (Network-Prefix) เพราะทุกโฮสต์ในเครือข่ายจะต้องมีพรีฟิกซ์หรือบิตนำหน้าเหมือนกัน ตัวอย่างเช่นหากมีเลขเครือข่ายจำนวน 16 บิต ก็จะเรียกว่า พรีฟิกซ์ 16 เป็นต้น

3.1.1 ความสำคัญของเลขเครือข่ายและเลขโฮสต์

การจัดแบ่งไอพีแอดเดรสออกเป็นสองส่วนที่ประกอบด้วยเลขเครือข่ายและเลขโฮสต์ก็เพื่อประโยชน์ในการดูแลระบบ เราเตอร์จะอาศัยเลขเครือข่ายเพื่อเลือกเส้นทางส่งแพ็กเก็ตด้วยหลักการต่อไปนี้

โฮสต์ที่มีเลขเครือข่ายชุดเดียวกันย่อมอยู่ภายในเครือข่ายเดียวกัน และสามารถสื่อสารถึงกันด้วยเฟรมดาต้าลิงค์โดยไม่ต้องพึ่งเราเตอร์

โฮสต์ที่มีเลขเครือข่ายต่างกันจะอยู่ต่างเครือข่ายกัน การสื่อสารระหว่างโฮสต์จะอาศัยเราเตอร์ที่เชื่อมต่อเครือข่ายเป็นผู้นำส่งแพ็กเก็ต เราเตอร์อาจเชื่อมเครือข่ายที่อยู่ติดกันหรือส่งแพ็กเก็ตผ่านเราเตอร์อื่นไปยังปลายทางดังรูป

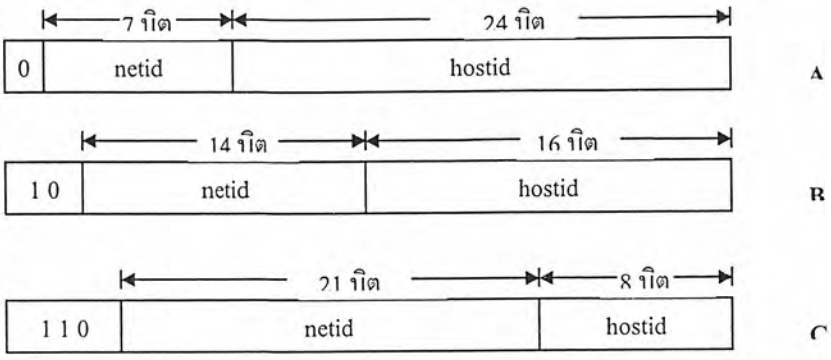


เครือข่าย	สถานี
1	1,2,3
2	1,2,3
3	1,2,3
4	1,2

รูปที่ 3-2 เราเตอร์เชื่อมโยงเครือข่ายที่มีเลขเครือข่ายต่างกัน

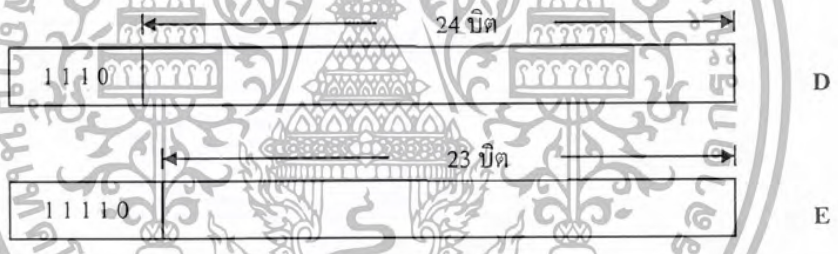
3.1.2 การจัดคลาสเครือข่าย

ไอพีแอดเดรสมีการจัดแบ่งออกเป็นกลุ่มหรือคลาส เครือข่ายที่ใช้งานในปัจจุบันมักสังกัดอยู่ในคลาสใดคลาสหนึ่งคือคลาส A, B หรือ C การแบ่งคลาสอาศัยจำนวนพรีฟิกซ์เครือข่ายที่แตกต่างกันตามรูปที่ 4-3 แต่ละคลาสจึงมีจำนวนเครือข่ายในสังกัดและจำนวนโฮสต์ต่อเครือข่ายไม่เท่ากัน



รูปที่ 3-3 การแบ่งคลาสเครือข่าย

การจัดคลาสตามรูปที่ 3-3 เป็นการจัดแบ่งตามการใช้งานเครือข่ายทั่วไป ในขณะที่ยังมีอีก 2 คลาสซึ่งใช้เพื่อจุดประสงค์เฉพาะได้แก่ คลาส D และ E ดังรูปที่ 4-4 เครือข่ายคลาส D เป็นเครือข่ายแบบมัลติคาสต์ซึ่งจะกล่าวในบทที่ 12 ส่วนคลาส E สงวนไว้ใช้งานหากมีความจำเป็นอื่นใดในอนาคต ทั้งสองคลาสนี้ไม่ได้แบ่งเลขโฮสต์จึงไม่กำหนดจำนวนโฮสต์ไว้



รูปที่ 3-4 การแบ่งคลาส D และ E

การจัดคลาสโดยใช้พรีฟิกซ์เป็นการผนวกข้อมูลเพื่อใช้ในการเลือกเส้นทาง เช่น หากตรวจพบว่าพรีฟิกซ์ 2 บิตแรกมีค่าเป็น 10 แสดงว่าเป็นแอดเดรสในคลาส B ซึ่งมีค่า 16 บิตแรกกำหนดกลุ่มเครือข่ายและ 16 บิตถัดมาเป็นเลขโฮสต์

3.1.3 ลักษณะสำคัญของแต่ละคลาส

จำนวนเครือข่ายในแต่ละคลาสและจำนวนโฮสต์สูงสุดที่มีได้ สามารถคำนวณได้จากจำนวนบิตที่ใช้งานตามสูตร 2^n เมื่อ n คือจำนวนบิต ตัวอย่างเช่นในคลาส B มีเลขโฮสต์จำนวน 16 บิต จึงมีโฮสต์ได้ไม่เกิน 2^{16} ซึ่งเท่ากับ 65,536 แต่เลขโฮสต์ที่ทุกบิตเป็น “0” และเป็น “1” จะสงวนไว้ใช้งานกรณีเฉพาะจำนวนโฮสต์จึงลดลงไป 2 โฮสต์ทุกเครือข่าย หรือมีโฮสต์ไม่เกิน $2^{16} - 2 = 65,534$ สูตร $2^n - 2$ นี้จะใช้กับการคำนวณจำนวนเครือข่ายในคลาสและจำนวนโฮสต์ ทั้งคลาส A, B และ C ดังนี้

คลาส A

เครือข่ายในคลาส A มีบิตซ้ายสุดเป็น 0 และใช้ 7 บิตถัดมากำหนดเครือข่าย ส่วนอีก 24 เป็นเลขโฮสต์ คลาส A จึงมีเลขเครือข่ายได้ 2^7 หรือ 128 ค่า แต่เครือข่าย 0.0.0.0 และ 127.0.0.0 สงวนไว้เป็นแอดเดรสเฉพาะงานคือ 0.0.0.0 เป็นแอดเดรสกำหนดเส้นทางโดยปริยาย (Default Route) ส่วน 127.0.0.0 เป็นแอดเดรสลูปแบ็คคือเป็นแอดเดรสที่ใช้เพื่อเชื่อมเข้าสู่อินเทอร์ลูปแบ็ค ดังนั้นจำนวนเครือข่ายในคลาส A จึงมีได้ 126 เครือข่ายคือเลขที่ขึ้นต้นด้วย 1.0.0.0 ถึง 126.0.0.0

แต่ละเครือข่ายในคลาส A มีแอดเดรสได้ $2^{24} - 2$ หรือเท่ากับ 16,777,214 คือตั้งแต่ 0.0.1 ถึง 255.255.254 เครือข่ายในคลาส A ใช้กับหน่วยงานขนาดใหญ่ที่ต้องการแอดเดรสเป็นจำนวนมาก เครือข่ายคลาสนี้จัดสรรให้กับหน่วยงานในยุคแรกเริ่มของอินเทอร์เน็ต แอดเดรสเครือข่ายที่เหลืออยู่ส่วนใหญ่จะสงวนไว้

สังเกตว่าในคลาส A นี้เมื่อกล่าวถึงเฉพาะเลขเครือข่ายก็จะเขียนเฉพาะค่าที่แสดงเลขเครือข่ายที่ขนาด 8 บิต เท่านั้นเช่น 2 หรือ 26 ในทำนองเดียวกันเมื่อกล่าวเฉพาะเลขโฮสต์ก็จะเขียนเฉพาะหมายเลขเครือข่ายโดยให้เลขโฮสต์เป็น "0" เช่น 2.0.0.0 รูปแบบการเขียนเช่นนี้ใช้กับคลาส B และ C เช่นกัน

คลาส B

เครือข่ายในคลาส B มีบิตแรกเริ่มเป็น 10 และใช้ 14 บิตถัดมากำหนดเลขเครือข่ายจำนวนบิตที่กำหนดเลขโฮสต์มีขนาด 16 บิต คลาส B จึงมีสมาชิกเครือข่ายได้ $2^{14} - 2$ หรือ 16,382 คือตั้งแต่ 128.1.0.0 ถึง 192.254.0.0 แต่ละเครือข่ายมีเลขโฮสต์ได้ $2^{16} - 2$ หรือเท่ากับ 65,534 แอดเดรส หรือตั้งแต่ 0.1 ถึง 255.254

เครือข่ายในคลาส B มักจัดสรรให้กับหน่วยงานขนาดกลาง ในปัจจุบันมีเครือข่ายในคลาส B เหลือไม่มากนัก และมักไม่จัดสรรเครือข่ายในคลาสนี้ให้กับผู้ขอจดทะเบียนรายใหม่หากไม่มีความจำเป็นอย่างแท้จริง

คลาส C

เครือข่ายในคลาส C มีพรีฟิกซ์ 110 และใช้ 21 บิตถัดมาเป็นเลขเครือข่าย จำนวนบิตที่เป็นเลขโฮสต์มีเพียง 8 บิต คลาส C จึงมีเลขเครือข่ายได้ตั้งแต่ 192.0.1.0 ถึง 223.255.254.0 รวม 2,097,150 เครือข่าย แต่ละเครือข่ายมีเลขโฮสต์ได้ตั้งแต่ 1 ถึง 254

จำนวนแอดเดรสได้จำกัดเพียง 254 แอดเดรสทำให้เครือข่ายเหมาะสำหรับหน่วยงานขนาดเล็ก หากจำเป็นต้องใช้โฮสต์มากกว่านี้ต้องขอใช้เครือข่ายคลาส C หลายเครือข่าย

คลาส D และ E

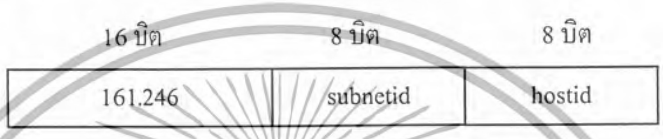
เครือข่ายในคลาส C และ D ไม่มีการจัดแบ่งเลขเครือข่ายและเลขโฮสต์ คลาส D มี 3 บิตแรกเป็น 111 จึงมีแอดเดรสตั้งแต่ 224.0.0.0 ถึง 239.255.255.255 แอดเดรสในคลาสนี้เรียกว่า มัลติคาสต์แอดเดรส(Multicast Address) เนื่องจากใช้ในเครือข่ายมัลติคาสต์

สำหรับคลาส E มีแอดเดรสจาก 240.0.0.0 ถึง 254.255.255.255 ซึ่งสำรองไว้เพื่อความจำเป็นเฉพาะงานในอนาคต

3.2 การแบ่งเครือข่ายย่อย

เครือข่ายที่สังกัดในคลาส A และ B เป็นเครือข่ายที่มีจำนวนโฮสต์ได้เป็นจำนวนมาก กล่าวคือ 16,777,214 และ 65,534 ตามลำดับ ในทางปฏิบัติแล้วเราไม่สามารถต่อเชื่อมโฮสต์ทั้งหมดในเครือข่ายเดี่ยวๆ ได้เพราะข้อจำกัดทางฮาร์ดแวร์ ผู้วางระบบจึงต้องจัดแบ่งเครือข่ายขนาดใหญ่ให้เล็กลงเป็นเครือข่ายขนาดย่อย หรือซับเน็ต (Subnet) การแบ่งซับเน็ต นอกจากจะจัดจำนวนโฮสต์ให้เหมาะสมกับฮาร์ดแวร์ของเครือข่ายแล้วยังช่วยอำนวยความสะดวกในการบริหารเครือข่าย

การจัดซับเน็ตใช้วิธีแบ่งบางส่วนของเลขโฮสต์มาใช้เป็นเลขซับเน็ต (SubnetID) เพื่อกำหนดว่าเป็นเครือข่ายย่อยที่เท่าใด ตัวอย่างเช่นเครือข่าย 161.246.0.0 ซึ่งอยู่ในคลาส B อาจใช้ 8 บิตแรกของเลขโฮสต์เป็นเลขซับเน็ต และ 8 บิตที่เหลือใช้สำหรับเลขโฮสต์ดังรูปที่ 4-5



รูปที่ 3-5 ตัวอย่างการแบ่งเครือข่ายย่อยของ 161.246

จำนวนบิตของเลขซับเน็ตเป็นตัวกำหนดจำนวนเครือข่ายย่อย ซับเน็ตขนาด 8 บิตสำหรับเครือข่าย 161.246.0.0 จะมี 254 ซับเน็ต ($2^{subnetid} - 2$) แต่ละซับเน็ตมี 254 โฮสต์ ($2^{hostid} - 2$) ดังตารางที่ 4.1 เลขซับเน็ตที่ทุกบิตเป็น "1" และ "0" จะสงวนไว้ใช้งานเฉพาะ ดังนั้นซับเน็ต 161.246.0.0 และ 161.246.255.0 จึงนำมาใช้ไม่ได้

ซับเน็ตที่	เครือข่ายย่อย	แอดเดรสเริ่มต้น	แอดเดรสสุดท้าย
1	161.246.1.0	161.246.1.1	161.246.1.254
2	161.246.2.0	161.246.2.1	161.246.2.254
3	161.246.3.0	161.246.3.1	161.246.3.254
..	..	..	..
..	..	..	..
252	161.246.252.0	161.246.252.1	161.246.252.254
253	161.246.253.0	161.246.253.1	161.246.253.254
254	161.246.254.0	161.246.254.1	161.246.254.254

ตารางที่ 3-1 การจัดแบ่งเครือข่าย 161.246 ด้วยซับเน็ต 8 บิต

3.2.1 ซับเน็ตมาสก์

เมื่อผู้วางระบบเลือกขนาดซับเน็ตแล้วจะกำหนดพารามิเตอร์เพื่อใช้บอกให้โฮสต์และเราเตอร์ทราบว่าซับเน็ตที่ใช้งานมีขนาดกี่บิต ค่านี้เรียกว่า ซับเน็ตมาสก์ (Subnet Mask)

ซับเน็ตมาสก์เป็นตัวเลข 32 บิต ซึ่งเขียนอยู่ในรูป Dotted-Decimal เช่นเดียวกับการเขียนไอพีแอดเดรส ซับเน็ตมาสก์จะมีบิตที่ตรงกับเลขเครือข่ายและเลขซับเน็ตเท่ากับ "1" ส่วนบิตที่ตรงกับเลยโฮสต์มีค่าเท่ากับ "0" การเลือกซับเน็ตมาสก์ควรใช้ค่าที่มีบิต "1" อยู่ติดกันจากทางซ้ายมือไปทางขวามือเสมอ

ตัวอย่างเครือข่าย 161.246.0.0 ซึ่งแบ่งให้มีเลขซับเน็ตและเลขโฮสต์อย่างละ 8 บิตจะมีค่าซับเน็ตมาสก์เท่ากับ 255.255.255.0 ค่านี้คำนวณได้จากการเขียนไอพีแอดเดรสทั้ง 4 หลัก และใส่เลขฐานสองค่า "1" ให้ครบทุกบิตที่เป็นเลขเครือข่ายและเลขซับเน็ต จากนั้นให้ใส่ค่า "0" สำหรับเลขโฮสต์ แล้วจึงแปลงเลขฐานสองที่

	8 บิต	8 บิต	8 บิต	8 บิต
1. นำค่าไอพีแอดเดรส	161	246	SubnetID	HostID
2. กำหนดบิต "1" และ "0"	11111111	11111111	11111111	00000000
3. แปลงเป็นเลขฐานสิบ	255	255	255	0

เครือข่าย 161.246.0.0 ซึ่งใช้ซับเน็ตมาสก์เท่ากับ 255.255.255.0 เรียกว่ามีซับเน็ตมาสก์ 24 บิต เนื่องจากมีบิตที่มีค่า "1" จำนวน 24 บิต หรือเขียนตามรูปแบบที่นิยมใช้ในปัจจุบันคือ 161.246.0.0/24 โดยเรียกว่าเครือข่าย 161.246.0.0 มีพรีฟิกซ์ 24 บิต

สังเกตว่า 161.246.0.0/24 ใช้เลขซับเน็ตจำนวน 8 บิต ดังนั้นนอกจากจะเรียกว่ามีพรีฟิกซ์ 24 บิตแล้ว ยังเรียกได้อีกว่าใช้ซับเน็ตบิตจำนวน 8 บิต

3.2.2 ดีฟอลต์ซับเน็ตมาสก์ (Default Subnet Mask)

การติดตั้งโฮสต์เข้าเครือข่ายนอกจากจะต้องกำหนดไอพีแอดเดรสแล้วต้องกำหนดค่าซับเน็ตมาสก์ตามที่ผู้ดูแลระบบกำหนดไว้ด้วย ถึงแม้ว่าในบางเครือข่ายเช่นเครือข่ายในคลาส C ซึ่งมีโฮสต์และไม่ได้แบ่งให้มีซับเน็ต ขั้นตอนการติดตั้งโฮสต์ยังจำเป็นต้องใส่ค่าซับเน็ตมาสก์เช่นกัน แต่ค่าซับเน็ตมาสก์นี้เรียกว่า ดีฟอลต์ซับเน็ตมาสก์ (Default Subnet Mask) ดีฟอลต์ซับเน็ตมาสก์ของเครือข่ายคลาส A, B และ C แสดงได้ดังตารางที่ 4.2

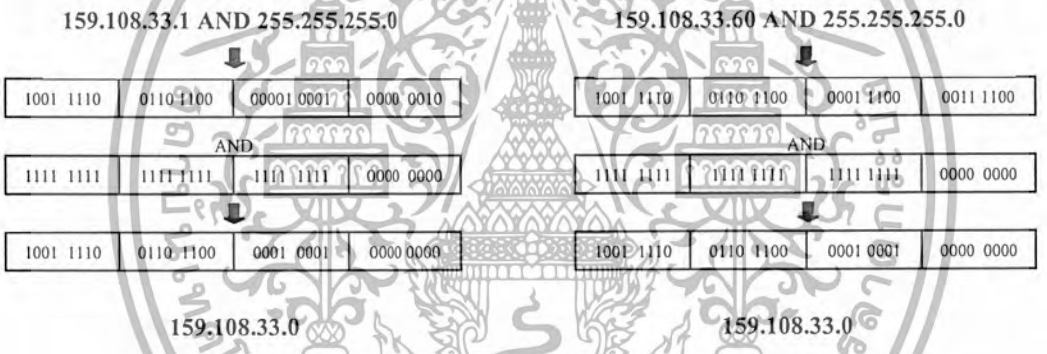
ผู้ดูแลระบบสามารถวางแผนจัดการเครือข่ายโดยเลือกทำซับเน็ตหรือไม่ทำซับเน็ตตามความต้องการ โดยปกติแล้วผู้ดูแลระบบเครือข่ายในคลาส A และ B ไม่สามารถหลีกเลี่ยงการใช้ซับเน็ตได้

คลาส	ดีฟอลต์ซับเน็ตมาสก์	ดีฟอลต์ซับเน็ตมาสก์(ฐาน 2)
A	255.0.0.0	11111111.00000000.00000000.00000000
B	255.255.0.0	11111111. 11111111.00000000.00000000
C	255.255.255.0	11111111.11111111.11111111.00000000

ตารางที่ 3-2 ค่าดีฟอลต์ซับเน็ตมาสก์

3.2.3 การเลือกเส้นทางในซับเน็ต

ซับเน็ตมาสก์นอกจากจะช่วยจัดแบ่งเครือข่ายย่อยแล้วยังใช้ประโยชน์ในการเลือกเส้นทางส่งไอพีดาต้าแกรมระหว่างเครือข่ายย่อยด้วย เช่น โฮสต์ 161.246.33.2 ในเครือข่าย 161.246.0.0/24 (ซับเน็ตมาสก์ 255.255.255.0) ต้องการส่งข้อมูลไปยังโฮสต์ 161.246.33.60 โพรโตคอลไอพีจะทำหน้าที่เลือกเส้นทางโดยนำแอดเดรส 161.246.33.2 และ 161.246.33.60 มาผ่านลอจิก "AND" บิตต่อบิตกับค่าซับเน็ตมาสก์ดังรูป



รูปที่ 3-6 การตรวจหาแอดเดรสซับเน็ตเพื่อเลือกเส้นทาง

ผลลัพธ์จากลอจิก "AND" ของแอดเดรสและเน็ตมาสก์ข้างต้นได้ค่าแอดเดรสซับเน็ต 161.246.33.0 เท่ากัน ซึ่งหมายความว่าโฮสต์ทั้งสองอยู่ในซับเน็ตเดียวกัน หากเครือข่ายที่ใช้คืออีเทอร์เน็ตแล้ว โฮสต์ 161.246.33.2 จะสร้างแพ็กเก็ตโดยระบุอีเทอร์เน็ตแอดเดรสของ 161.246.33.60 โดยไม่ต้องส่งแพ็กเก็ตให้เราเตอร์ดำเนินการ โปรดสังเกตว่าการใช้ลอจิก "AND" เป็นการใช้ซับเน็ตมาสก์เพื่อ "มาสก์" ให้ได้เฉพาะเลขเครือข่าย ค่าซับเน็ตมาสก์จึงเป็นเสมือน หน้ากาก ครอบเอาเลขเครือข่ายออกมา

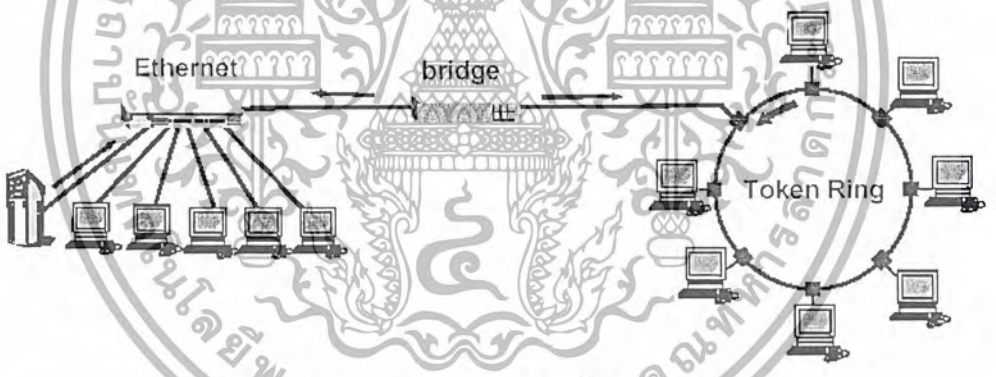
ในกรณีที่โฮสต์ปลายทางอยู่ต่างเครือข่ายกับโฮสต์ต้นทาง เช่น แอดเดรสของโฮสต์ต้นทางคือ 161.246.33.2 และโฮสต์ปลางทางคือ 161.246.40.5 ผลจากลอจิก "AND" ระหว่าง 161.246.40.5 กับมาสก์ 255.255.255.0 จะได้ค่า 158.0108.40.0 ซึ่งต่างจาก 161.246.33.0 ดังนั้นโฮสต์ 161.246.33.2 จะสรุปว่า 161.246.40.0 ซึ่งต่างจาก 161.246.33.0 ดังนั้นโฮสต์ 161.246.33.2 จะสรุปว่า 161.246.40.0 อยู่ต่างซับเน็ต และจะส่งแพ็กเก็ตไปยังเราเตอร์เพื่อให้เราเตอร์เพื่อให้เราเตอร์นำส่งแพ็กเก็ตต่อไป

บทที่ 4
พื้นฐานบริดจ์และสวิตช์

4.1 บริดจ์ (Bridge)

ในองค์กรขนาดใหญ่มักจะมีระบบเครือข่ายแลนอยู่หลายระบบ ซึ่งอาศัยการเชื่อมต่อโดยใช้เราเตอร์ (Router) สำหรับเครือข่ายที่ใช้โพรโตคอลอย่างเดียวกันเท่านั้น อุปกรณ์เครือข่ายประเภทบริดจ์ จึงได้รับการพัฒนาขึ้นมา เพื่อใช้ในการเชื่อมต่อระบบเครือข่ายต่างชนิดกันเข้าด้วยกัน โดยอาศัยแมคแอดเดรสในการกำหนดเส้นทางการสื่อสาร ดังนั้นบริดจ์จึงสามารถเรียกอีกอย่างได้ว่า Low-level router

เนื่องจากบริดจ์ทำงานในชั้นที่ 2 (Data Link Layer) ดังนั้นจึงมองไม่เห็นความแตกต่างของแพ็กเก็ต IP, IPX และอื่น ๆ ทำให้สามารถรับ-ส่งข้อมูลของโพรโตคอลได้เกือบทุกชนิด แต่การควบคุมเส้นทางในการส่งข้อมูลของบริดจ์ จะมีความยืดหยุ่นน้อยกว่าเราเตอร์ เนื่องจากจะใช้ข้อมูลแมคแอดเดรสเท่านั้นในการกำหนดเส้นทาง ดังนั้น บริดจ์จึงเหมาะสมกับระบบเครือข่ายที่มีความซับซ้อนไม่มากนัก



รูปที่ 4-1 แสดงระบบเครือข่ายที่ใช้บริดจ์ในการเชื่อมต่อ

4.1.1 ชนิดของบริดจ์

บริดจ์มีอยู่ 2 ชนิด คือ ทรานส์แพเร้นท์บริดจ์ (Transparent Bridge) และ ซอสเราท์บริดจ์ (Source Route Bridge)

4.1.1.1 ทรานส์แพเร้นท์บริดจ์ (Transparent Bridge)

พัฒนาโดย Digital Equipment Corporation ในต้นปี 1980 ซึ่งต่อมาได้มีการกำหนดมาตรฐาน IEEE 802.1 โดยผู้ใช้งานได้กำหนดความต้องการทรานส์แพเร้นท์บริดจ์ให้มีลักษณะดังนี้ แรกทีเดียวผู้ใช้ (ทุกระบบ) จะต้องไม่มีส่วนเกี่ยวข้องกับการทำงานของทรานส์แพเร้นท์บริดจ์ การมีอุปกรณ์ประเภทนี้อยู่ในระบบ หรือไม่

มีอยู่ก็ตาม จะต้องไม่มีผลกระทบใด ๆ ต่อผู้ใช้งาน ผู้ใช้ทั่วไปจะต้องสามารถซื้ออุปกรณ์นี้มาจากตัวแทนจำหน่ายของบริษัทใดก็ได้ การติดตั้งจะต้องมีความยุ่งยากเพียงแค่การเสียบปลั๊กไฟฟ้าและเสียบสายระบบเครือข่ายต่าง ๆ เข้ากับอุปกรณ์นี้ แม้ว่าจะไม่มีการกำหนดค่าพารามิเตอร์ใด ๆ ตัวอุปกรณ์ก็จะต้องสามารถทำงานได้ในทันที ผลที่เกิดขึ้นนั้นน่าแปลกใจเป็นอย่างยิ่งว่า ทรานส์แพเร้นท์บริดจ์นั้นมีตัวตนและทำงานได้อย่างที่ผู้ใช้ต้องการ

ทรานส์แพเร้นท์บริดจ์ ส่วนใหญ่จะใช้ในการเชื่อมต่อระหว่างเซกเมนต์ (Segment) ของอีเทอร์เน็ต โดยการทำงานของทรานส์แพเร้นท์บริดจ์ จะเก็บข้อมูลแมคแอดเดรสของสเตชันที่ต่ออยู่ของพอร์ตต่าง ๆ โดยจะใช้ข้อมูลนั้น เมื่อมีเฟรมส่งเข้ามาที่พอร์ตนั้น

MAC Address Table

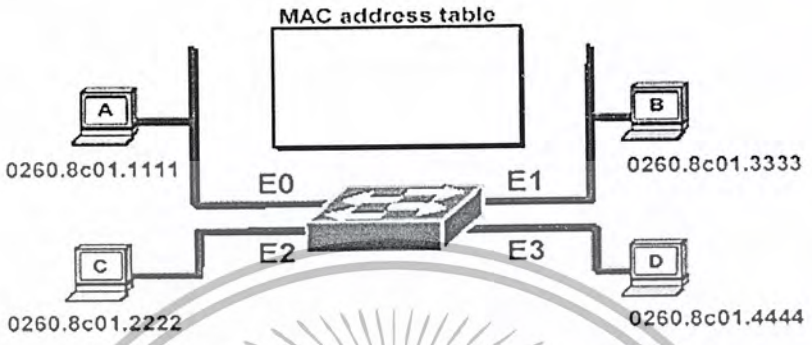
E0: 0260.8c01.1111
E2: 0260.8c01.2222
E1: 0260.8c01.3333
E3: 0260.8c01.4444

รูปที่ 4-2 แสดงตัวอย่าง MAC Address Table

เมื่อบริดจ์ได้รับเฟรมเข้ามา จะตรวจสอบแอดเดรสของสเตชันปลายทางและส่งเฟรมนั้น ออกไปยังพอร์ตที่ระบุ ยกเว้นเฟรมที่มีแอดเดรสของสเตชั่นปลายทางเป็นพอร์ทเดียวกันกับสเตชั่นต้นทาง แต่ละรายการในตารางจะมีเวลากำหนดไว้เรียกว่า Time-To-Live (TTL) โดยรายการนั้นจะถูกลบออกจากตารางเมื่อถึงเวลา TTL ที่กำหนดและ TTL จะถูกกำหนดใหม่เมื่อมีเฟรมจากสเตชันของรายการนั้นเข้ามาอีกครั้ง ซึ่งจะช่วยแก้ปัญหาในกรณีที่มีการย้ายสเตชันไปยังพอร์ตอื่น หรือการนำสเตชันนั้นออกจากระบบเครือข่าย ในกรณีที่บริดจ์ไม่สามารถหารายการที่ตรงกับแอดเดรสของสเตชั่นปลายทางได้ เฟรมนั้นจะถูกส่งไปยังทุกพอร์ตยกเว้นพอร์ตที่รับเฟรมเข้ามา

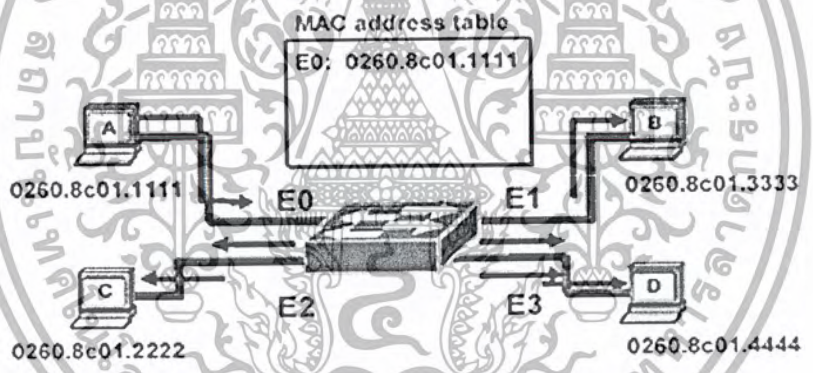
ทรานส์แพเร้นท์บริดจ์จะช่วยให้สามารถแบ่งระบบเครือข่ายออกเป็นเซกเมนต์ย่อย ๆ เพื่อลดปริมาณของการส่งข้อมูลในรูปแบบอีเทอร์เน็ตในอุปกรณ์ฮับไม่ให้คับคั่งมากเกินไป

ตัวอย่างการส่งข้อมูลและการสร้างตารางแมคแอดเดรส



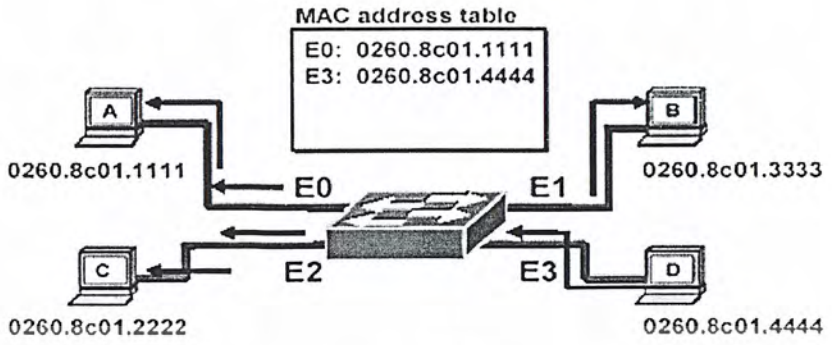
รูปที่ 4-3 แสดงการเรียนรู้แมคแอดเดรส

จากรูป เป็นตอนเริ่มต้น ยังไม่มีการส่งข้อมูล ดังนั้น ในตารางแมคแอดเดรสจึงยังว่างเปล่าอยู่ (empty)



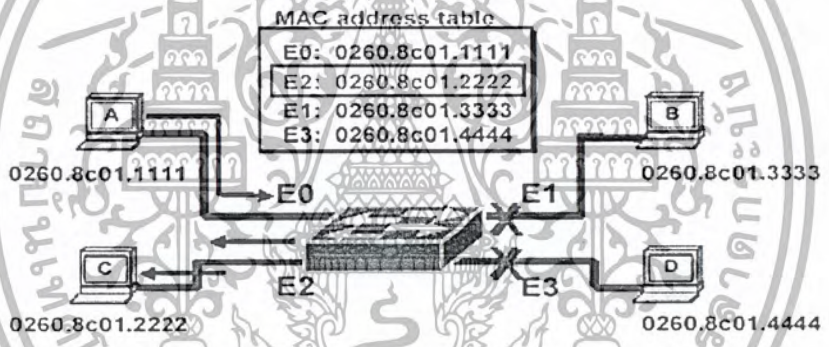
รูปที่ 4-4 แสดงการเรียนรู้แมคแอดเดรส

จากรูปสเตชัน A (Station A) ต้องการส่งข้อมูลไปยังสเตชัน C (Station C) ดังนั้นที่เฟรมข้อมูลของเอจะมีแอดเดรสของสเตชันต้นทาง (Source Address) เป็นแมคแอดเดรสของ A คือ 0260.8c01.1111 และแอดเดรสของสเตชันปลายทาง (Destination Address) เป็นแมคแอดเดรสของ C คือ 0260.8c01.2222 เมื่อสเตชัน A ส่งเฟรมไปยังสวิตช์ จะทำการดูที่สเตชันต้นทางทำการเรียนรู้ว่า พอร์ต E0 ที่รับข้อมูลเข้ามาคือ 0260.8c01.1111 จากนั้นทำการใส่ลงในตารางแมคแอดเดรสเป็น E0: 0260.8c01.1111 จากนั้นดูต่อไปที่สเตชันปลายทาง แล้วจึงไปค้นหาที่ตารางแมคแอดเดรสว่ามีแมคแอดเดรสนี้อยู่หรือไม่ จากรูป เราจะเห็นได้ว่า ในตารางไม่มีสเตชันปลายทางอยู่ เมื่อเป็นในกรณีนี้ สวิตช์จะทำการส่งออกไปยังทุกพอร์ต (Flood Out)



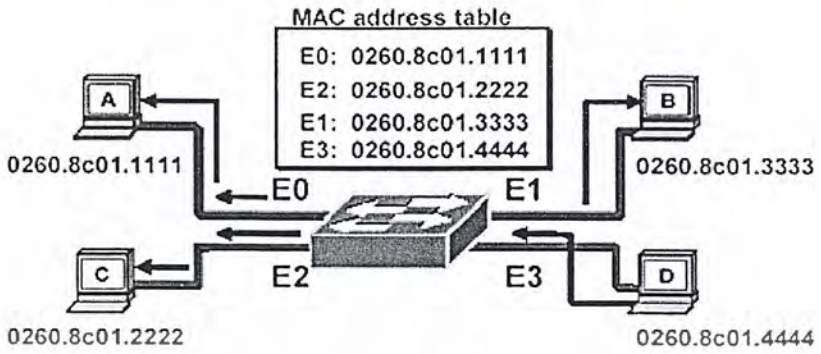
รูปที่ 4-5 แสดงการเรียนรู้แมคแอดเดรส

จากรูป สเตชัน D (Station D) ต้องการส่งเฟรมข้อมูลไปยังสเตชัน C (Station C) จะเป็นกรณีเดียวกับในรูป 4-4 นั่นเอง



รูปที่ 4-6 แสดงการทำงานของสวิตช์ในการทำฟิลเตอร์ (Filter)

จากรูปสเตชัน A ต้องการส่งเฟรมข้อมูลไปยังสเตชัน C เมื่อสวิตช์ทำการเรียนรู้แมคแอดเดรส แล้วจึงทำการค้นหาในตารางและพบว่าสเตชันปลายทางมีจุดหมายอยู่ที่พอร์ต E2 จึงทำการส่งเฟรมข้อมูลไปยังพอร์ต E2 เพียงพอร์ตเดียว ไม่ส่งไปยังพอร์ตอื่น ๆ



รูปที่ 4-7 แสดงการเรียนรู้แมคแอดเดรส

เมื่อสเตชัน D ต้องการส่งเฟรมข้อมูลแบบบอร์ดคาสต์ (Broadcast) หรือ เมาติคาสต์ (Multicast) สวิตช์ก็จะทำการส่งข้อมูลออกไปทุกพอร์ตยกเว้นพอร์ต E3 ซึ่งก็คือ พอร์ตที่รับเฟรมเข้ามานั่นเอง

4.1.1.2 ซอสเราท์บริดจ์ (Source-Route Bridge (SRB))

SRB เป็นอัลกอริธึมที่พัฒนาโดย IBM สำหรับการเชื่อมต่อระหว่างแลนแบบโทเก้นริง (IEEE 802.5) โดยการส่งข้อมูลแบบ SRB จะต้องมีการกำหนดเส้นทางก่อนล่วงหน้า ซึ่งมีขั้นตอนในการหาเส้นทาง คือ

เมื่อ โฮสต์เอ็กซ์ (Host X) ต้องการส่งเฟรมให้โฮสต์วาย (Host Y) ในครั้งแรก โฮสต์เอ็กซ์จะไม่ทราบว่าโฮสต์วายอยู่ในเครือข่ายแลนเดียวกันหรือไม่ โฮสต์เอ็กซ์จะทำการส่งเฟรมทดสอบ ถ้าเฟรมกลับมายังโฮสต์เอ็กซ์ โดยที่บิต A ในเฟรมโทเก้นริงไม่เป็น 1 แสดงว่า โฮสต์วายอยู่ต่างเซกเมนต์

จากนั้น โฮสต์เอ็กซ์จะทำการส่งเฟรมเอกโพลเรอร์ (Explorer) ไปยังบริดจ์ บริดจ์เมื่อได้รับเฟรมเอกโพลเรอร์ จะส่งเฟรมนั้นไปยังทุกพอร์ตยกเว้นพอร์ตที่รับเฟรม โดยเพิ่มข้อมูลของเส้นทาง (Route) ในเฟรมเอกโพลเรอร์ตามบริดจ์นั้น เมื่อเฟรมเอกโพลเรอร์ไปถึงโฮสต์วายจะทำการตอบกลับมายังโฮสต์เอ็กซ์ตามข้อมูลเส้นทาง ซึ่งอาจจะตอบกลับมาหลายครั้งตามจำนวนเฟรมเอกโพลเรอร์ที่ได้รับ

ซึ่งโฮสต์เอ็กซ์จะต้องเลือกเส้นทางใดเส้นทางหนึ่ง โดยที่วิธีในการเลือกนั้นไม่ได้กำหนดไว้ใน IEEE 802.5 แต่สามารถเป็นไปได้หลายอย่าง เช่น

- เลือกเฟรมตอบกลับแรกที่ได้รับ
- เลือกเฟรมที่มีฮอป (Hop) น้อยที่สุด
- เลือกเส้นทางที่ยอมให้มีเฟรมขนาดใหญ่ที่สุด

เป็นต้น

เมื่อเลือกเส้นทางได้แล้ว ข้อมูลเส้นทางจะถูกกำหนดลงในเฟรมที่จะส่งไปที่โฮสต์วายในรูปแบบ Routing Information Field (RIF)

4.1.2 การเปรียบเทียบบริดจ์ในระบบ 802

บริดจ์ทั้งแบบทรานส์แพเร้นท์บริดจ์และแบบ SRB มีทั้งข้อดีและข้อเสียที่แตกต่างกันดังที่สรุปไว้ในตารางในรูปที่ 4-8

Issue	Transparent bridge	Source routing bridge
Orientation	Connectionless	Connection-oriented
Transparency	Fully transparent	Not transparent
Configuration	Automatic	Manual
Routing	Suboptimal	Optimal
Locating	Backward learning	Discovery frames
Failures	Handled by the bridges	Handled by the hosts
Complexity	In the bridges	In the hosts

ตารางที่ 4-1 ตารางเปรียบเทียบคุณสมบัติของทรานส์แพเร้นท์บริดจ์และแบบ SRB

หัวใจของความแตกต่างระหว่างบริดจ์ทั้ง 2 ชนิดคือ การสื่อสารเครือข่ายแบบมีการติดต่อช่วงสั้น (Connectionless) และแบบมีการติดต่ออย่างต่อเนื่อง (Connection-oriented) ทรานส์แพเร้นท์บริดจ์ไม่ใช้แนวคิดของวงจรเสมือน เส้นทางเดินของแต่ละเฟรมจะถูกเลือกอย่างเป็นอิสระ ส่วน SRB มีลักษณะตรงกันข้ามคือต้องมีการค้นหาเส้นทางเดินข้อมูลให้ได้เสียก่อน จากนั้นจึงใช้เส้นทางที่ค้นพบสำหรับการส่งข้อมูลจริงในภายหลัง

การทำงานของทรานส์แพเร้นท์บริดจ์จะไม่เข้าไปเกี่ยวข้องกับโฮสต์เลยแม้แต่น้อย และยังสามารถทำงานเข้ากันได้กับการส่งข้อมูลตามมาตรฐาน 802 ทุกระบบ ในขณะที่ SRB ต้องให้โฮสต์เข้ามาร่วมกระบวนการทำงานด้วยและไม่สามารถทำงานร่วมกับการส่งข้อมูลตามมาตรฐาน 802 บางระบบได้ นั่นคือโฮสต์จะต้องรู้จักโครงสร้างและการทำงานของบริดจ์เป็นอย่างดี และสามารถทำงานร่วมกันได้ การแบ่งระบบเครือข่ายออกเป็น 2 วงที่เชื่อมกันโดยวิธีการเลือกทางเดินโดยผู้ส่งข้อมูลจะต้องทำการเปลี่ยนแปลงโปรแกรมของโฮสต์ด้วย

การใช้ทรานส์แพเร้นท์บริดจ์ไม่จำเป็นต้องมีระบบบริหารเครือข่าย บริดจ์สามารถเรียนรู้ได้ด้วยตัวเองและสามารถปรับตัวให้เข้ากับระบบเครือข่ายนั้น ๆ ได้โดยอัตโนมัติ ส่วน SRB จะต้องให้ผู้บริหารเครือข่ายทำการติดตั้งหมายเลขระบบเครือข่ายและหมายเขบริดจ์ด้วยตนเองทั้งหมด ความผิดพลาดเช่น ระบบเครือข่ายหรือบริดจ์ใช้หมายเลขซ้ำกันนั้นตรวจสอบได้ยากมาก ซึ่งอาจจะทำให้เกิดการรวบซ้ำของเฟรมข้อมูลได้ นอกจากนี้การติดต่อของระบบเครือข่าย 2 แห่งที่เคยเชื่อมต่อกันในอดีตนั้น สำหรับทรานส์แพเร้นท์บริดจ์แล้วไม่มีสิ่งใดต้องทำยกเว้นการเชื่อมต่อสายเข้าด้วยกันเท่านั้น ส่วนการใช้ SRB อาจมีความจำเป็นจะต้องเปลี่ยนหมายเลขของระบบเครือข่ายหลาย ๆ ระบบที่ใช้หมายเลขซ้ำกัน

ข้อดีประการหนึ่งของ SRB อยู่ที่ระบบนี้สามารถใช้เส้นทางการส่งข้อมูลที่ดีที่สุดได้ (ในทางทฤษฎี) ในขณะที่ทรานส์แพเร้นท์บริดจ์มีข้อจำกัดที่เกิดขึ้นจากการใช้สแปนนิ่งทรี นอกจากนี้ SRB สามารถเลือกใช้งานบริดจ์คู่ขนานระหว่างระบบเครือข่ายได้อย่างเหมาะสม แต่บริดจ์ที่ใช้งานจริงจะฉลาดมากพอที่จะแบ่งงานกันทำให้ได้ตามที่กล่าวไว้หรือไม่นั้น ยังไม่มีการพิสูจน์

การค้นหาตำแหน่งของผู้รับโดยวิธีการเรียนรู้ย้อนหลังในทรานส์แพเร้นท์บริดจ์มีข้อจำกัดตรงที่บริดจ์จะต้องรอจนกระทั่งเฟรมที่ส่งมาจากสถานีต่าง ๆ นั้นมาถึงจึงจะสามารถเรียนรู้จากข้อมูลเหล่านั้นได้ ส่วนการค้นหาเฟรมใน SRB มีปัญหาเกี่ยวกับการเพิ่มจำนวนของเฟรมค้นหาอย่างรวดเร็วมาก โดยเฉพาะในระบบที่มีจำนวนระบบเครือข่ายมากและใช้บริดจ์คู่เชื่อมต่อระหว่างเครือข่ายเข้าด้วยกัน

การจัดการความผิดพลาดของบริดจ์ทั้งสองแบบมีวิธีการที่แตกต่างกัน ในแบบแรกบริดจ์สามารถเรียนรู้เกี่ยวกับบริดจ์และระบบเครือข่ายต่าง ๆ ที่ทำงานผิดพลาดหรือการเปลี่ยนแปลงรูปแบบเครือข่ายได้อย่างรวดเร็วและเป็นไปอย่างอัตโนมัติ ด้วยการดักฟังสัญญาณที่ส่งออกมาจากอุปกรณ์เหล่านั้นเพียงอย่างเดียว โฮสต์จะไม่ต้องเข้ามายุ่งเกี่ยวด้วยเลย

ส่วนการจัดการความผิดพลาดในระบบของ SRB มีวิธีการที่แตกต่างออกไปอย่างสิ้นเชิง เมื่อบริดจ์หยุดทำงาน สเตชันที่เลือกเส้นทางเดินข้อมูลผ่านอุปกรณ์ตัวนั้นจะพบว่าเฟรมที่ส่งออกไปไม่ได้รับการตอบรับกลับมาเลย สเตชันนั้นอาจส่งข้อมูลซ้ำแล้วซ้ำอีก สุดท้ายสเตชันนั้นจะทราบว่าเกิดมีอุปกรณ์บางอย่างทำงานผิดปกติ แต่ก็ยังไม่ทราบว่ามีปัญหาเกิดขึ้นที่สเตชันปลายทางหรืออยู่ในเส้นทางปัจจุบัน การหาคำตอบด้วการส่งเฟรมค้นหาข้อมูลออกไปใหม่จะทำให้ทราบว่าสเตชันปลายทางยังคงทำงานอยู่หรือไม่ อย่างไรก็ตาม ในกรณีที่บริดจ์หลักเสียหายหรือหยุดทำงานจะทำให้โฮสต์จำนวนมากเสียเวลาไปกับการรอคอยและส่งเฟรมค้นหาออกไปจนกว่าปัญหานั้นจะได้รับการแก้ไขแม้ว่าจะมีเส้นทางอื่นอยู่ก็ตาม การชำรุดของอุปกรณ์เป็นจุดอ่อนหลักของการเชื่อมต่อแบบต่อเนื่องทั้งหมด

4.2 สวิตช์ (Switch)

สวิตช์เป็นอุปกรณ์ในระบบเครือข่ายที่ออกแบบมาเพื่อแยกระบบเครือข่ายออกเป็นส่วนย่อย ๆ เพื่อเพิ่มประสิทธิภาพของระบบเครือข่ายและทำให้การควบคุมระบบเครือข่ายทำได้ดีขึ้น โดยแต่ละพอร์ตของสวิตช์จะเป็นเซกเมนต์หนึ่งของระบบเครือข่าย ข้อมูลที่ส่งในเซกเมนต์เดียวกันจะไม่ถูกส่งไปยังเซกเมนต์อื่น เป็นการช่วยลดปัญหาความคับคั่งของข้อมูลได้

สวิตช์จะมีลักษณะคล้ายกับบริดจ์ในการแบ่งระบบเครือข่ายออกเป็นส่วนย่อย ๆ ในกรณีที่มีการส่งข้อมูลข้ามเซกเมนต์ สวิตช์จะส่งเฟรมไปยังพอร์ตที่สเตชันปลายทางอยู่เท่านั้น อีกทั้งสวิตช์ยังสามารถส่งข้อมูลระหว่างเซกเมนต์ได้พร้อม ๆ กัน โดยไม่เกิดปัญหาการชนกันของข้อมูล (Collision) และสามารถส่งข้อมูลได้ในแบบสองทาง (Full-duplex)

สวิตช์จะทำงานที่ชั้น 2 (Data link Layer) ของ OSI Reference Model (โดยในปัจจุบัน สวิตช์สามารถทำงานได้ที่ชั้นที่ 2 ชั้นที่ 3 และชั้นที่ 4 แล้ว) ดังนั้นสวิตช์จะรับและส่งเฟรมข้อมูลตาม MAC Address ของสเตชันที่ต่ออยู่ที่พอร์ตของสวิตช์ โดยการต่อเชื่อมกับสวิตช์แบ่งออกเป็น 2 แบบคือ Segment switch และ Port switch

Segment Switch จะรองรับทราฟฟิกของสเตชันในเซกเมนต์ในแต่ละพอร์ต รวมทั้งเซกเมนต์ที่มีสเตชันเดียวด้วย ซึ่งจะเชื่อมต่อจากสเตชันมาที่พอร์ตของสวิตช์โดยตรง ซึ่งทำให้ผู้ออกแบบระบบเครือข่ายสามารถจัดให้สเตชั่นที่ต้องมีการส่งข้อมูลกันมาก ๆ หรือบ่อย ๆ อยู่ในเซกเมนต์เดียวกัน และสามารถจัดให้เซิร์ฟเวอร์ที่ให้บริการ

Port Switch หรือเรียกอีกอย่างหนึ่งว่า Switch Hub เป็นการใช้งานในลักษณะ 1 พอร์ตต่อ 1 สเตชัน โดยใช้งานแทนที่ฮับ

4.2.1 คัททรูสวิตช์ (Cut-Through Switching)

โดยการทำงานปกติของสวิตช์ จะทำการรับเฟรมเข้ามาก่อน แล้วจึงส่งเฟรมนั้นไปยังพอร์ตของสเตชันปลายทาง (Store & Forward) สวิตช์แบบคัททรูจะลดการหน่วงเวลาในขั้นตอนนี้ โดยเมื่อสวิตช์ได้รับข้อมูลเฟรมเพียงพอที่จะกำหนดสเตชั่นเป้าหมายได้แล้ว ก็จะเริ่มต้นการส่งข้อมูลทันทีโดยไม่ต้องรอให้ได้รับเฟรมทั้งหมด

การใช้งานสวิตช์แบบคัททรูอาจจะทำให้เกิดปัญหาการส่งเฟรมที่มีข้อผิดพลาดได้ ดังนั้นจึงควรกำหนดให้สวิตช์ทำการรับเฟรมมาจำนวนหนึ่งก่อน แล้วจึงเริ่มการส่งเฟรม เพื่อให้แน่ใจว่า เฟรมนั้นเป็นเฟรมที่ไม่มีข้อผิดพลาด

4.2.2 ชนิดของสวิตช์

Crossbar Switch เป็นสวิตช์ที่พัฒนาขึ้นในยุคแรก ๆ โดยทุกอินพุตจะต่อเข้ากับทุก ๆ เอาต์พุต โดยจะมีบัฟเฟอร์ของอินพุตที่ใช้ในการพักข้อมูลเมื่อพอร์ตเอาต์พุตกำลังใช้งานอยู่

Shared-memory Switch สวิตช์ชนิดนี้จะเก็บข้อมูลที่รับเข้าไว้ในหน่วยความจำและส่งออกไปยังพอร์ตของสเตชั่นปลายทาง ข้อมูลจะเข้าและออกระหว่างพอร์ตกับหน่วยความจำโดยตรง วิธีการนี้มีข้อเสียคือ เกิดความล่าช้าในการเก็บข้อมูลลงในหน่วยความจำ

High-speed bus Switch ข้อมูลที่เข้ามาที่พอร์ตจะส่งผ่านบัสและส่งออกไปยังพอร์ตที่สเตชั่นปลายทางเชื่อมต่ออยู่ บัสที่ใช้จะเป็นบัสความเร็วสูง โดยใช้เทคนิค TDM ในการให้บริการกับพอร์ตต่าง ๆ ซึ่งจะต้องมีบัฟเฟอร์ที่ใช้ในการพักข้อมูลไว้ชั่วคราว

สวิตช์แบบ High-speed bus เป็นชนิดที่มีการนำมาใช้มากที่สุด เช่น สวิตช์รุ่น Catalyst 3000 ของซิสโก้ (Cisco) ที่มีพอร์ต 10Base-T 16 พอร์ตและรองรับการเชื่อมต่อแบบฟาสต์อีเทอร์เน็ต (Fast Ethernet), ATM หรือแวน (WAN) จะใช้บัสความเร็ว 480 Mbps และมีบัฟเฟอร์ขนาด 256 K โดยใช้ชิปโพรเซสเซอร์ Intel i960 ในการควบคุมการเข้าถึงบัสของแต่ละพอร์ต

4.2.3 สวิตช์เลเยอร์ที่ 3 (Layer 3 Switch)

สวิตช์เลเยอร์ที่ 3 หรือ L3 Switch คือสวิตช์ที่ทำงานในระดับชั้นที่ 3 (Network Layer) ดังนั้นการเลือกเส้นทางการส่งข้อมูลของสวิตช์ L3 จึงต้องอาศัยข้อมูลที่อยู่ในแพ็กเก็ตของชั้นที่ 3 เช่นเดียวกับเราเตอร์ นอกจากนี้ยังต้องทำหน้าที่อื่น ๆ ที่กำหนดในชั้นที่ 3 ด้วย เช่น การตรวจสอบความถูกต้องของข้อมูลโดยการ checksum การตรวจสอบการหมดอายุของแพ็กเก็ต (TTL) การรองรับโพรโตคอลการจัดการต่าง ๆ ของชั้นที่ 3 และระบบควบคุมการปลอดภัย

Characteristic	Layer 3 Switch	Router
LAN Protocol (IP, IPX, Apple Talk)	Yes	Yes
Subnet definition	Layer 2 Switch domain	Port
Forwarding architecture	Hardware	Software (ASIC)
Management	SNMP, RMON	SNMP (RMON)
WAN support	No	Yes
Price	Low	High

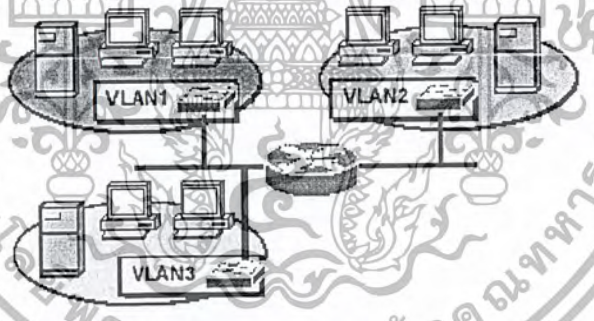
ASIC – Application Specific Integrated Circuit

ตารางที่ 4-2 แสดงตารางการเปรียบเทียบระหว่างสวิตช์ชั้นที่ 2 กับสวิตช์ชั้นที่ 3

บทที่ 5
แลนเสมือน

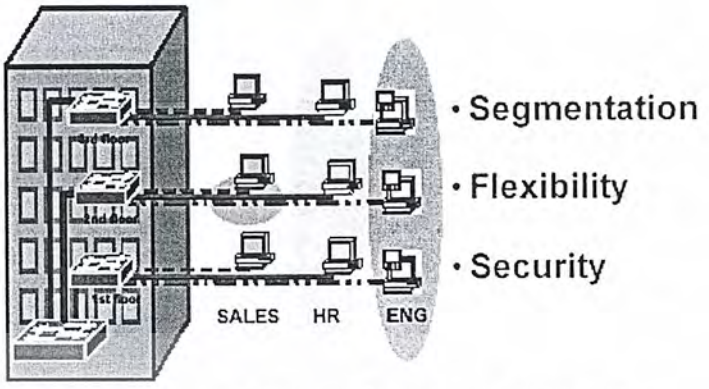
เมื่อพิจารณาถึงระบบเครือข่ายที่ประกอบด้วยอุปกรณ์ในชั้นที่ 2 เท่านั้น เช่น เซกเมนต์ของอีเทอร์เน็ต, สวิตช์ที่มีหลายพอร์ต หรือเครือข่ายที่ประกอบไปด้วยสวิตช์หลาย ๆ ตัว เครือข่ายแบบนี้ เรียกว่า Flat Network Topology เครือข่ายแบบนี้จะมีได้เพียง 1 บอร์ดคาสต์โดเมน (Broadcast Domain) เท่านั้น หมายความว่า เมื่อมีการส่งเฟรมแบบบอร์ดคาสต์จะทำให้ทุก ๆ สเตชันได้รับเฟรมนี้ไป ดังนั้นยิ่งมีจำนวนของอุปกรณ์ (เช่น สวิตช์, ฮับ, สเตชัน) มากขึ้นเท่าใด ก็ยิ่งทำให้เกิดทราฟฟิกขึ้นในเครือข่ายมากขึ้นเท่านั้น จนอาจทำให้เกิดเป็นบอร์ดคาสต์สตอร์ม (Broadcast Storm) ได้

แลนเสมือน หรือวีแลน (Virtual LANs (VLANs)) คือการสร้างเซกเมนต์ของระบบเครือข่ายที่ไม่ขึ้นกับระบบเครือข่ายทางกายภาพ หมายความว่าเราสามารถแบ่งเครือข่ายเราออกเป็นเครือข่ายย่อย ๆ ได้ โดยไม่ขึ้นต่อกัน เมื่อกำหนดวีแลนขึ้นมาแล้ว เราจะถือว่า แต่ละวีแลนเป็น 1 บอร์ดคาสต์โดเมนเป็นเครือข่ายแลนที่ไม่เกี่ยวข้องต่อกัน



รูปที่ 5-1 แสดงเครือข่ายวีแลน

เมื่อกำหนดวีแลนแล้ว สเตชันในวีแลนเดียวกันจะสามารถส่งข้อมูลถึงกันได้ แต่ถ้าเป็นการส่งข้อมูลข้ามเซกเมนต์จะต้องใช้เราเตอร์ในการส่งผ่านข้อมูล ซึ่งผู้ดูแลระบบสามารถกำหนดรายละเอียดของการส่งผ่านข้อมูลระหว่างเซกเมนต์ได้



รูปที่ 5-2 แสดงตัวอย่างการแบ่งวีแลนออกเป็นแผนกงาน

วีแลนมีข้อดีคือ

1.*Segmentation* คือสามารถแบ่งเครือข่ายออกเป็นเครือข่ายย่อยได้ เป็นการแบ่งทราฟฟิกของแต่ละวีแลนออกจากัน

2.*Flexibility* คือ มีความยืดหยุ่นในการเปลี่ยนแปลงสมาชิกที่อยู่ในวีแลนได้ง่าย

3.*Security* คือ เมื่อเราทำการแบ่งวีแลนแล้ว จะถือว่าแต่ละวีแลนเป็น 1 บอร์ดคาสต์โดเมนทำให้การส่งข้อมูลไม่รั่วไหลออกไปยังวีแลนอื่น ๆ เป็นข้อมูลที่ส่งอยู่ในวีแลนเดียวเท่านั้น

5.1 ประเภทของวีแลน

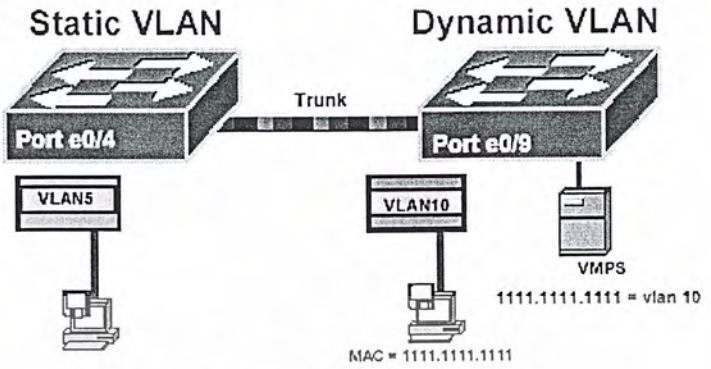
แบ่งออกเป็น 2 ประเภทใหญ่ ๆ คือ

5.1.1 สแตติกวีแลน (Static VLANs)

เป็นการกำหนดวีแลนจากพอร์ตของสวิตช์ว่า ต้องการให้พอร์ตไหนเป็นวีแลนใด เมื่อเรานำอุปกรณ์ไปต่อ ก็จะทำให้อุปกรณ์ชิ้นนั้นเป็นสมาชิกของวีแลนนั้นโดยอัตโนมัติ มีข้อดีคือ กำหนดได้ง่าย และดูแลง่าย (Based on port)

5.1.2 ไดนามิกวีแลน (Dynamic VLANs)

เป็นการกำหนดวีแลนตามค่าแมคแอดเดรสที่ได้กำหนดไว้ โดยจะมีฐานข้อมูล (Database) เก็บไว้ว่าแมคแอดเดรสค่าใดเป็นสมาชิกของวีแลนใด จะมีข้อดีเมื่อเราทำการเคลื่อนย้ายอุปกรณ์ใด ๆ ก็จะยังทำให้อุปกรณ์ตัวนั้นเป็นสมาชิกของวีแลนเดิมอยู่โดยอัตโนมัติ ไม่จำเป็นต้องไปกำหนดค่าใหม่ (Based on MAC Address)



รูปที่ 5-3 แสดงประเภทของวีแลน

นอกจากนี้ เรายังสามารถแบ่งประเภทวีแลนออกได้เป็นอีกหลายแบบ คือ

1. **Port-Based & MAC-Based**

   Port-Based : กำหนดวีแลนตามพอร์ตที่กำหนดไว้ (เหมือน Static VLAN)

   MAC-Based : กำหนดวีแลนตามแมคแอดเดรสที่กำหนดไว้ (เหมือน Dynamic VLAN)

2. **Protocol-Based & Dynamic-Based**

   Protocol-Based : กำหนด วีแลน ตามโพรโตคอลที่กำหนดไว้ เช่น

   Host X ใช้โพรโตคอล IP ดังนั้น จะเป็นสมาชิกของวีแลน 1

   Host Y ใช้โพรโตคอล IPX ดังนั้น จะเป็นสมาชิกของวีแลน 2

   Dynamic-Based : กำหนดวีแลนตาม User Profile ที่กำหนดไว้ โดยเก็บ User Profile ไว้ในฐานข้อมูล เช่น โฮสต์ X ทำการ Log in ตัวโพรไฟล์ (Profile) ของ โฮสต์ X จะเป็นตัวกำหนดให้โฮสต์ X เป็นของ วีแลน 1

5.2 ประเภทของการเชื่อมต่อ

ในการกำหนดวีแลนนั้น บางครั้งอาจมีการกำหนดให้ในวีแลนเดียวกันมีอุปกรณ์ที่เป็นสมาชิกอยู่ในสวิตช์คนละตัวกัน ดังนั้นจึงต้องมีการกำหนดการเชื่อมต่อของวีแลนเพื่อใช้เป็นกฎในการส่งข้อมูลภายในวีแลนเดียวกัน

5.2.1 **แอ็กเซสลิงค์ (Access Link)** เป็นการเชื่อมต่อที่บอกว่าลิงค์ (Link) นี้เป็นวีแลนใด โดยข้อมูลที่ผ่านจะมีแค่วีแลนเดียว

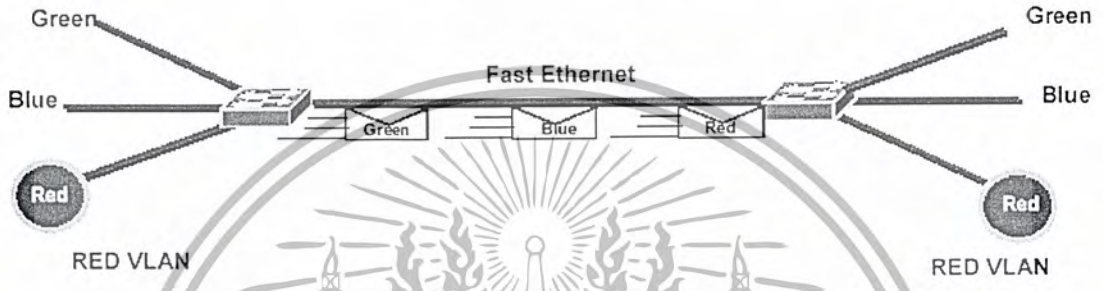
5.2.2 **ทรังค์ลิงค์ (Trunk Link)** เป็นการเชื่อมต่อที่ใช้ในการส่งข้อมูลได้หลายๆ วีแลน

รูปที่ 5-4 แสดงตัวอย่างของทรังค์ลิงค์

5.3 วิธีการระบุถึงวีแลน

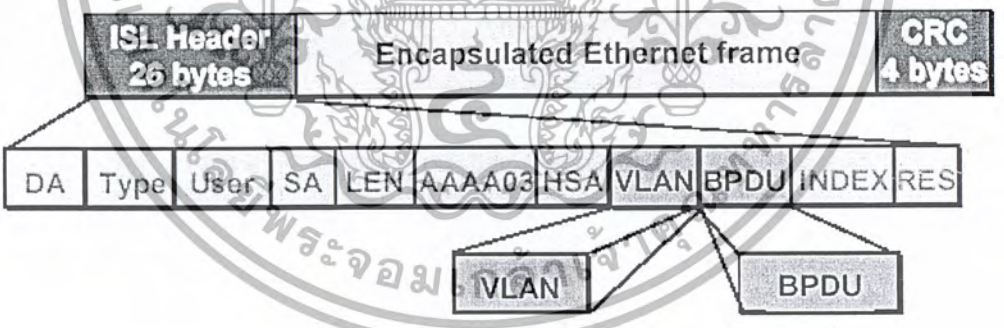
แพ็กเก็ตจะถูกส่งไปตาม Trunk Link โดยบรรจุข้อมูลที่ระบุถึงวีแลนไว้ในส่วนของเฮดเดอร์ (Header) ของแพ็กเก็ต ซึ่งวิธีการระบุนั้นมีอยู่ 2 แบบคือ

- Cisco ISL
- IEEE 802.1Q



รูปที่ 5-5 แสดงเครือข่ายของวีแลนที่ใช้การเชื่อมต่อแบบทรังค์ลิงก์

Cisco ISL



รูปที่ 5-6 แสดงรูปแบบของเฟรมของ ISL

เมื่อได้รับเฟรมอีเทอร์เน็ตมาแล้ว จะทำการ encapsulate ด้วย ISL Header และ CRC ซึ่งสนับสนุนวีแลนได้มากสุดถึง 1024 วีแลน

IEEE 802.1Q

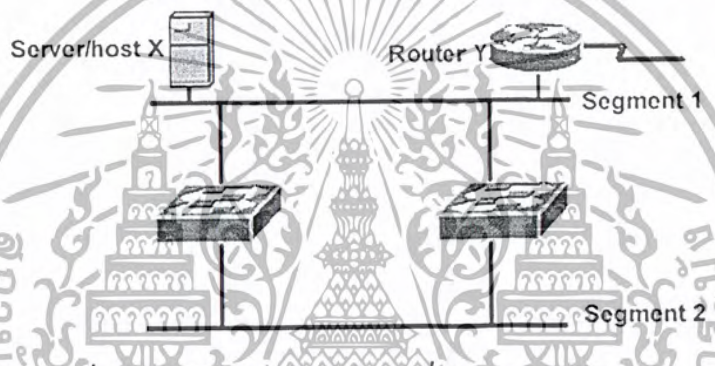
Initial MAC Address	2-Byte TPID 2-Byte TCI	Initial Type/Data	New CRC

รูปที่ 5-7 แสดงรูปแบบของเฟรมของ IEEE 802.1Q

- 2-byte tag protocol identifier (TPID)
- 2-byte tag control information (TCI)

บทที่ 6
สแปนนิ่งทรีโพรโตคอล

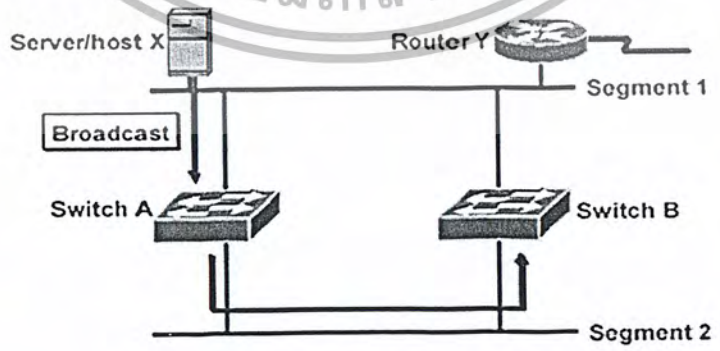
ในการสร้างระบบเครือข่ายที่ดีนั้น ควรต้องคำนึงถึงความน่าจะเป็นในทุกด้าน เมื่อเกิดกรณีฉุกเฉินระบบเครือข่ายจะทำการแก้ไขเช่นไร เช่น ถ้าในระบบ สวิตช์เกิดเสีย ใช้งานไม่ได้ จะทำเช่นไร ด้วยเหตุนี้ จึงทำให้เกิดเป็นแนวคิด Redundant Topology ขึ้น คือเป็นการสร้างระบบเครือข่ายแบบต่อให้มี สวิตช์ 2 ตัวในเส้นทาง (path) เดียวกัน เพื่อป้องกันในเหตุที่ถ้ามีสวิตช์ตัวใดตัวหนึ่งเสีย สวิตช์ตัวที่เหลือจะยังสามารถใช้งานต่อไปได้ ทำให้ระบบเครือข่ายไม่หยุดชะงัก



รูปที่ 6-1 แสดงตัวอย่างของเครือข่ายที่ใช้ Redundant Topology

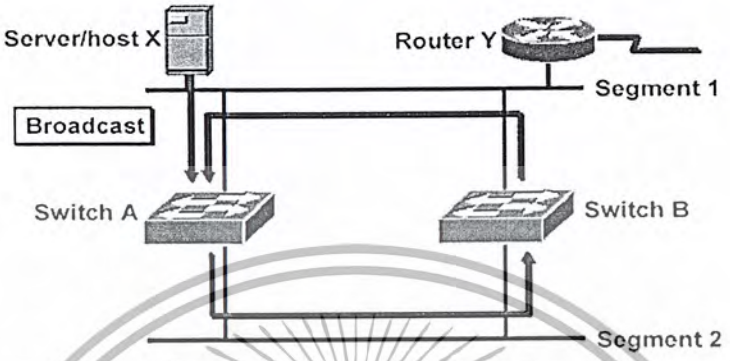
เมื่อเราสร้างระบบเครือข่ายโดยใช้ Redundant Topology แล้ว อาจทำให้เกิดลูป (loop) ขึ้นได้ ซึ่งเป็นสาเหตุทำให้เกิดบอร์ดคาสต์สตอร์มขึ้นในระบบเครือข่ายจนอาจเป็นสาเหตุทำให้เครือข่ายล่มได้

โดยขั้นตอนของการเกิดลูปจะเป็นดังนี้คือ



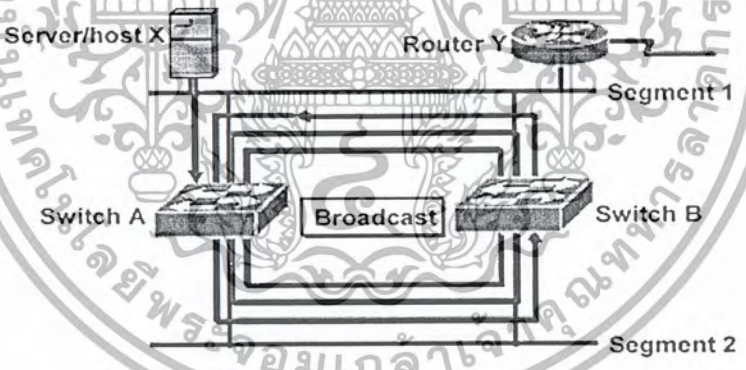
รูปที่ 6-2 แสดงการส่งข้อมูลจากโฮสต์ X ไปยังสวิตช์ A

เมื่อโฮสต์ X ส่งเฟรมข้อมูลออกไปให้กับสวิตช์ A จากนั้นสวิตช์ A จะทำการใส่แอดเดรสต้นทาง (Source Address) เข้าไปในตารางแมคแอดเดรสแล้วจึงทำการส่งออกไปยังทุกพอร์ตออกไป ทำให้สวิตช์ B ได้รับเฟรมข้อมูลนี้ด้วย



รูปที่ 6-3 แสดงการฟลัดเอาท์ (Flood out) ของสวิตช์ A

เมื่อสวิตช์ B ได้รับเฟรมข้อมูล ก็จะทำการใส่แอดเดรสต้นทางลงไปในตารางแมคแอดเดรสแล้วทำการส่งออกไปยังทุกพอร์ต (Flood Out) อีกครั้ง ทำให้สวิตช์ A ได้รับเฟรมข้อมูลนี้อีกครั้ง



รูปที่ 6-4 แสดงการเกิดบอร์ดคาสต์สตอร์ม (Broadcast Storm)

เมื่อสวิตช์ A ได้รับเฟรมข้อมูล ก็จะกระทำการเหมือนเดิม และต่อเนื่องไปถึงสวิตช์ B วนเป็นลูปไปเรื่อย ๆ ไม่รู้จบ และแม้ว่า เฟรมข้อมูลไปถึง ณ จุดหมายที่สเตชันที่ต้องการแล้ว ตัวสวิตช์ A และสวิตช์ B ก็จะยังส่งเฟรมข้อมูลกันต่อไปแบบไม่รู้จบ จนทำให้เกิดเป็นบอร์ดคาสต์สตอร์ม สร้างทราฟฟิกให้กับระบบเครือข่าย จนอาจทำให้เครือข่ายหยุดชะงักได้

6.1 สแปนนิ่งทรีโพรโตคอล (Spanning-Tree Protocol)

สแปนนิ่งโพรโตคอลถูกสร้างมาเพื่อใช้ในการกำจัดลูปที่เกิดขึ้นในระบบเครือข่าย โดยจะเสมือนจัดให้ระบบเครือข่ายมีลักษณะเหมือนกับต้นไม้ (Tree) คือจากจุดเริ่มต้นแล้วมีการแตกกิ่งก้านออกมาเรื่อย ๆ เป็นสายแห่งการติดต่อสื่อสาร จะทำให้เป็นเครือข่ายที่ไม่มีลูปเลย

หลักการทำงานของสแปนนิ่งโพรโตคอลคือ สวิตช์ทุกตัวจะทำการส่งเฟรมชื่อว่า Bridge Protocol Data Unit (BPDU) ออกไปยังทุกพอร์ต เพื่อแสดงถึงความมีตัวตนของสวิตช์ แล้วสวิตช์ทุกตัวจะได้รับ BPDU จากสวิตช์ตัวข้างเคียงเพื่อนำมาใช้ในการคำนวณโดยใช้สแปนนิ่งอัลกอริธึม (Spanning Tree Algorithm) ซึ่งมีหลักการคือ

- เลือกรูทบริดจ์ (Root Bridge)
- เลือกรูทพอร์ต (Root Port)
- เลือกดีไซเนเตทพอร์ต (Designated Port)

สวิตช์ทุกตัวจะทำการส่ง BPDU ออกไปยังทุกพอร์ต เป็นการแสดงถึงความมีตัวตนของสวิตช์ จากนั้นสวิตช์ทุกตัวจึงจะได้รับ BPDU มาเพื่อทำการคำนวณ โดยใช้สแปนนิ่งอัลกอริธึมจะได้เป็นสแปนนิ่งทรีโพรเซส (Spanning-Tree Process) ซึ่งก็คือ

6.1.1 เลือกรูทบริดจ์

Root Bridge คือจุดที่เป็นจุดอ้างอิง (Reference) ของสแปนนิ่งทรี ซึ่งก็คือ จุดยอดของต้นไม้นั่นเอง โดยรูทบริดจ์จะเป็นสวิตช์ตัวที่มีบริดจ์ไอดี (Bridge ID) น้อยที่สุด

บริดจ์ไอดีประกอบด้วย

- Bridge Priority (2 bytes) ไพรออริตี้ (Priority) หรือ Weight ของสวิตช์สามารถมีแค่ได้ตั้งแต่ 0 – 65535 และมีค่าดีฟอลต์ (default) คือ 32768
- แมคแอดเดรส (8 bytes) เป็นสิ่งที่แสดงให้เห็นถึงความเป็นเอกของแต่ละสวิตช์ เนื่องจากแมคแอดเดรสมีคุณสมบัติคือ มีเพียงแมคแอดเดรสเดียวในโลก ไม่เหมือนใคร และไม่สามารถเปลี่ยนแปลงได้

ในตอนเริ่มต้นนั้น สวิตช์จะตั้งค่ารูทบริดจ์เป็นบริดจ์ไอดีตัวเองก่อน จากนั้นจึงมีการรับ BPDU จากสวิตช์ข้างเคียง แล้วนำมาคำนวณหาตัวที่น้อยกว่า ทำเช่นนี้ไปเรื่อย ๆ จนกว่า สวิตช์ทุกตัวจะมี Root Bridge เป็นตัวเดียวกัน และหลังจากนั้นสวิตช์ก็ยังจะส่ง BPDU ทุก ๆ 2 วินาที (เป็นค่าดีฟอลต์)

6.1.2 เลือกรูทพอร์ต

สำหรับทุก ๆ นอนรูทบริดจ์ (Non-root Bridge) คือสวิตช์ที่ไม่ใช่รูทบริดจ์ต้องทำการเลือกรูทพอร์ต โดยเลือกพอร์ตที่ดีที่สุดในการสื่อสารไปยังรูทบริดจ์ โดยคำนวณจากพอร์ตคอสท์ (Port cost) หรือรูทพาร์ธคอสท์ (Root path cost – จำนวนฮอปทั้งหมดจากรูทบริดจ์จนถึงสวิตช์)

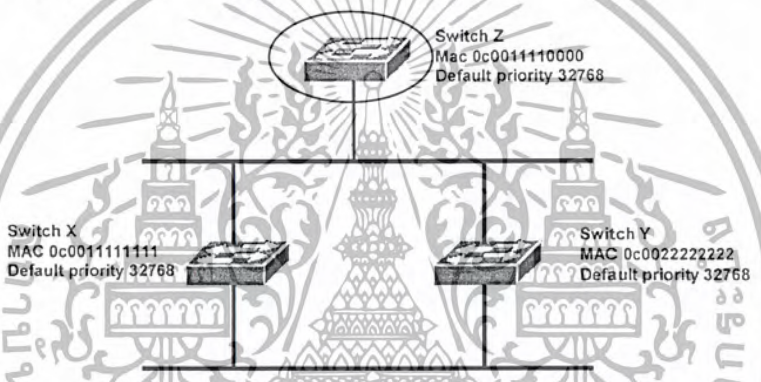
6.1.3 เลือกดีไซเนเตทพอร์ต

ดีไซเนเตทพอร์ตมีคอนเซ็ปต์ว่า ลิงค์เพียงหนึ่งเดียวในเซกเมนต์ที่ใช้ในการส่งและรับทราฟ ฟิก โดยสำหรับตัวรูทบริดจ์นั้นถือว่า ทุก ๆ พอร์ตของรูทบริดจ์จะเป็นดีไซเนเตทพอร์ตและสำหรับนอนรูทบริดจ์จะให้พอร์ตที่ต่อกับรูทพอร์ตของสวิตช์ตัวข้างเคียงเป็นดีไซเนเตทพอร์ต และสำหรับพอร์ตที่ไม่ใช่รูทพอร์ตและดีไซเนเตทพอร์ตจะถูกบล็อก (Block)

ด้วยหลักการทำงานนี้ จะทำให้แต่ละเครือข่ายแลนสามารถส่งเฟรมข้อมูลไปยังสวิตช์ได้เพียงเครื่องเดียว และเกิดสภาพของต้นไม้ (Tree) ขึ้น โดยพอร์ตที่ไม่ได้ใช้งานจะเป็นพอร์ตสำรอง และเมื่อดำเนินการทำจนเกิดเป็นสแปนนิ่งทรีแล้ว จะได้เครือข่ายที่มีลักษณะ

- **One Root Bridge per Network**

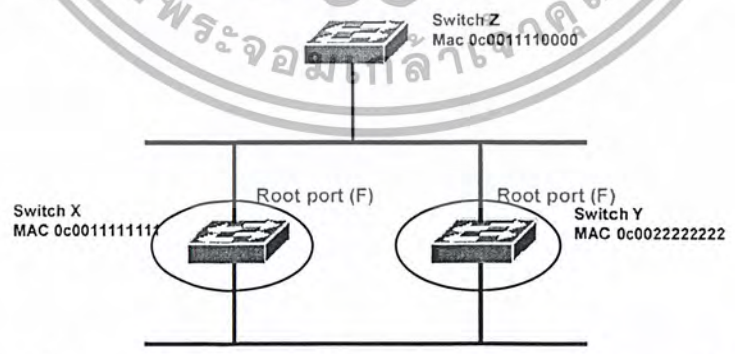
  1 Root Bridge ต่อ 1 ระบบเครือข่าย



รูปที่ 6-5 แสดงระบบเครือข่ายที่มีสวิตช์ Z เป็นรูทบริดจ์

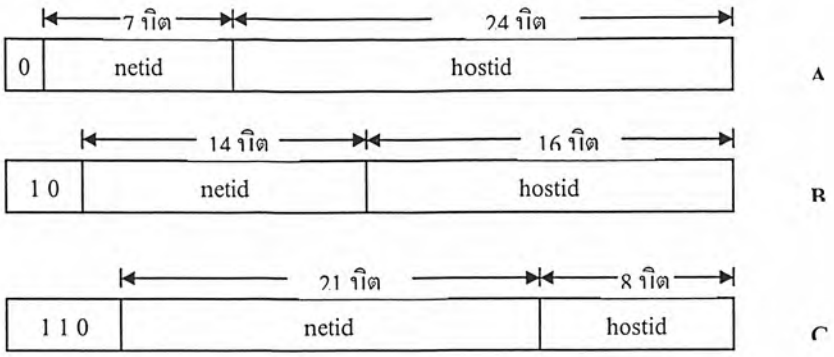
- **One Root port per Non-Root Bridge**

  1 Root Port ต่อ 1 Non-root Bridge



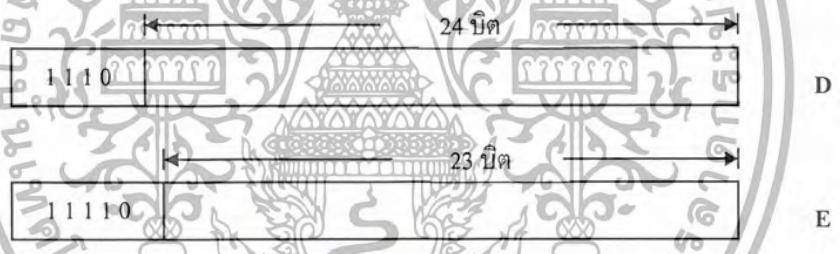
รูปที่ 6-6 แสดงระบบเครือข่ายที่มีสวิตช์ X และสวิตช์ Y เป็นนอนรูทบริดจ์

- **One Designated port per Segment**



รูปที่ 3-3 การแบ่งคลาสเครือข่าย

การจัดคลาสตามรูปที่ 3-3 เป็นการจัดแบ่งตามการใช้งานเครือข่ายทั่วไป ในขณะที่ยังมีอีก 2 คลาสซึ่งใช้เพื่อจุดประสงค์เฉพาะได้แก่ คลาส D และ E ดังรูปที่ 4-4 เครือข่ายคลาส D เป็นเครือข่ายแบบมัลติคาสต์ซึ่งจะกล่าวในบทที่ 12 ส่วนคลาส E สงวนไว้ใช้งานหากมีความจำเป็นอื่นใดในอนาคต ทั้งสองคลาสนี้ไม่ได้แบ่งเลขโฮสต์จึงไม่กำหนดจำนวนโฮสต์ไว้



รูปที่ 3-4 การแบ่งคลาส D และ E

การจัดคลาสโดยใช้พรีฟิกซ์เป็นการผนวกข้อมูลเพื่อใช้ในการเลือกเส้นทาง เช่น หากตรวจพบว่าพรีฟิกซ์ 2 บิตแรกมีค่าเป็น 10 แสดงว่าเป็นแอดเดรสในคลาส B ซึ่งมีค่า 16 บิตแรกกำหนดกลุ่มเครือข่ายและ 16 บิตถัดมาเป็นเลขโฮสต์

3.1.3 ลักษณะสำคัญของแต่ละคลาส

จำนวนเครือข่ายในแต่ละคลาสและจำนวนโฮสต์สูงสุดที่มีได้ สามารถคำนวณได้จากจำนวนบิตที่ใช้งานตามสูตร 2^n เมื่อ n คือจำนวนบิต ตัวอย่างเช่นในคลาส B มีเลขโฮสต์จำนวน 16 บิต จึงมีโฮสต์ได้ไม่เกิน 2^{16} ซึ่งเท่ากับ 65,536 แต่เลขโฮสต์ที่ทุกบิตเป็น "0" และเป็น "1" จะสงวนไว้ใช้งานกรณีเฉพาะจำนวนโฮสต์จึงลดลงไป 2 โฮสต์ทุกเครือข่าย หรือมีโฮสต์ไม่เกิน $2^{16} - 2 = 65,534$ สูตร $2^n - 2$ นี้จะใช้กับการคำนวณจำนวนเครือข่ายในคลาสและจำนวนโฮสต์ ทั้งคลาส A, B และ C ดังนี้

คลาส A

เครือข่ายในคลาส A มีบิตซ้ายสุดเป็น 0 และใช้ 7 บิตถัดมากำหนดเครือข่าย ส่วนอีก 24 เป็นเลขโฮสต์ คลาส A จึงมีเลขเครือข่ายได้ 2^7 หรือ 128 ค่า แต่เครือข่าย 0.0.0.0 และ 127.0.0.0 สงวนไว้เป็นแอดเดรสเฉพาะงานคือ 0.0.0.0 เป็นแอดเดรสกำหนดเส้นทางโดยปริยาย (Default Route) ส่วน 127.0.0.0 เป็นแอดเดรสลูปแบ็คคือเป็นแอดเดรสที่ใช้เพื่อเชื่อมเข้าสู่อินเทอร์ลูปแบ็ค ดังนั้นจำนวนเครือข่ายในคลาส A จึงมีได้ 126 เครือข่ายคือเลขที่ขึ้นต้นด้วย 1.0.0.0 ถึง 126.0.0.0

แต่ละเครือข่ายในคลาส A มีแอดเดรสได้ $2^{24} - 2$ หรือเท่ากับ 16,777,214 คือตั้งแต่ 0.0.1 ถึง 255.255.254 เครือข่ายในคลาส A ใช้กับหน่วยงานขนาดใหญ่ที่ต้องการแอดเดรสเป็นจำนวนมาก เครือข่ายคลาสนี้จัดสรรให้กับหน่วยงานในยุคแรกเริ่มของอินเทอร์เน็ต แอดเดรสเครือข่ายที่เหลืออยู่ส่วนใหญ่จะสงวนไว้

สังเกตว่าในคลาส A นี้เมื่อกล่าวถึงเฉพาะเลขเครือข่ายก็จะเขียนเฉพาะค่าที่แสดงเลขเครือข่ายที่ขนาด 8 บิต เท่านั้นเช่น 2 หรือ 26 ในทำนองเดียวกันเมื่อกล่าวเฉพาะเลขโฮสต์ก็จะเขียนเฉพาะหมายเลขเครือข่ายโดยให้เลขโฮสต์เป็น "0" เช่น 2.0.0.0 รูปแบบการเขียนเช่นนี้ใช้กับคลาส B และ C เช่นกัน

คลาส B

เครือข่ายในคลาส B มีบิตแรกเริ่มเป็น 10 และใช้ 14 บิตถัดมากำหนดเลขเครือข่ายจำนวนบิตที่กำหนดเลขโฮสต์มีขนาด 16 บิต คลาส B จึงมีสมาชิกเครือข่ายได้ $2^{14} - 2$ หรือ 16,382 คือตั้งแต่ 128.1.0.0 ถึง 192.254.0.0 แต่ละเครือข่ายมีเลขโฮสต์ได้ $2^{16} - 2$ หรือเท่ากับ 65,534 แอดเดรส หรือตั้งแต่ 0.1 ถึง 255.254

เครือข่ายในคลาส B มักจัดสรรให้กับหน่วยงานขนาดกลาง ในปัจจุบันมีเครือข่ายในคลาส B เหลือไม่มากนัก และมักไม่จัดสรรเครือข่ายในคลาสนี้ให้กับผู้ขอจดทะเบียนรายใหม่หากไม่มีความจำเป็นอย่างแท้จริง

คลาส C

เครือข่ายในคลาส C มีพรีฟิกซ์ 110 และใช้ 21 บิตถัดมาเป็นเลขเครือข่าย จำนวนบิตที่เป็นเลขโฮสต์มีเพียง 8 บิต คลาส C จึงมีเลขเครือข่ายได้ตั้งแต่ 192.0.1.0 ถึง 223.255.254.0 รวม 2,097,150 เครือข่าย แต่ละเครือข่ายมีเลขโฮสต์ได้ตั้งแต่ 1 ถึง 254

จำนวนแอดเดรสได้จำกัดเพียง 254 แอดเดรสทำให้เครือข่ายเหมาะสำหรับหน่วยงานขนาดเล็ก หากจำเป็นต้องใช้โฮสต์มากกว่านี้ต้องขอใช้เครือข่ายคลาส C หลายเครือข่าย

คลาส D และ E

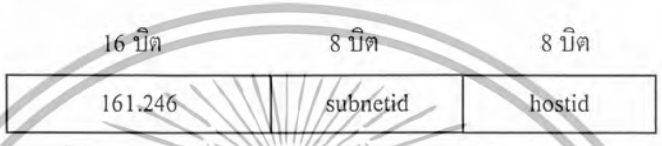
เครือข่ายในคลาส C และ D ไม่มีการจัดแบ่งเลขเครือข่ายและเลขโฮสต์ คลาส D มี 3 บิตแรกเป็น 111 จึงมีแอดเดรสตั้งแต่ 224.0.0.0 ถึง 239.255.255.255 แอดเดรสในคลาสนี้เรียกว่า มัลติคาสต์แอดเดรส(Multicast Address) เนื่องจากใช้ในเครือข่ายมัลติคาสต์

สำหรับคลาส E มีแอดเดรสจาก 240.0.0.0 ถึง 254.255.255.255 ซึ่งสำรองไว้เพื่อความจำเป็นเฉพาะงานในอนาคต

3.2 การแบ่งเครือข่ายย่อย

เครือข่ายที่สังกัดในคลาส A และ B เป็นเครือข่ายที่มีจำนวนโฮสต์ได้เป็นจำนวนมาก กล่าวคือ 16,777,214 และ 65,534 ตามลำดับ ในทางปฏิบัติแล้วเราไม่สามารถต่อเชื่อมโฮสต์ทั้งหมดในเครือข่ายเดี่ยวๆ ได้เพราะข้อจำกัดทางฮาร์ดแวร์ ผู้วางระบบจึงต้องจัดแบ่งเครือข่ายขนาดใหญ่ให้เล็กลงเป็นเครือข่ายขนาดย่อย หรือซับเน็ต (Subnet) การแบ่งซับเน็ต นอกจากจะจัดจำนวนโฮสต์ให้เหมาะสมกับฮาร์ดแวร์ของเครือข่ายแล้วยังช่วยอำนวยความสะดวกในการบริหารเครือข่าย

การจัดซับเน็ตใช้วิธีแบ่งบางส่วนของเลขโฮสต์มาใช้เป็นเลขซับเน็ต (SubnetID) เพื่อกำหนดว่าเป็นเครือข่ายย่อยที่เท่าใด ตัวอย่างเช่นเครือข่าย 161.246.0.0 ซึ่งอยู่ในคลาส B อาจใช้ 8 บิตแรกของเลขโฮสต์เป็นเลขซับเน็ต และ 8 บิตที่เหลือใช้สำหรับเลขโฮสต์ดังรูปที่ 4-5



รูปที่ 3-5 ตัวอย่างการแบ่งเครือข่ายย่อยของ 161.246

จำนวนบิตของเลขซับเน็ตเป็นตัวกำหนดจำนวนเครือข่ายย่อย ซับเน็ตขนาด 8 บิตสำหรับเครือข่าย 161.246.0.0 จะมี 254 ซับเน็ต ($2^{subnetid} - 2$) แต่ละซับเน็ตมี 254 โฮสต์ ($2^{hostid} - 2$) ดังตารางที่ 4.1 เลขซับเน็ตที่ทุกบิตเป็น "1" และ "0" จะสงวนไว้ใช้งานเฉพาะ ดังนั้นซับเน็ต 161.246.0.0 และ 161.246.255.0 จึงนำมาใช้ไม่ได้

ซับเน็ตที่	เครือข่ายย่อย	แอดเดรสเริ่มต้น	แอดเดรสสุดท้าย
1	161.246.1.0	161.246.1.1	161.246.1.254
2	161.246.2.0	161.246.2.1	161.246.2.254
3	161.246.3.0	161.246.3.1	161.246.3.254
..	..	..	..
..	..	..	..
252	161.246.252.0	161.246.252.1	161.246.252.254
253	161.246.253.0	161.246.253.1	161.246.253.254
254	161.246.254.0	161.246.254.1	161.246.254.254

ตารางที่ 3-1 การจัดแบ่งเครือข่าย 161.246 ด้วยซับเน็ต 8 บิต

3.2.1 ซับเน็ตมาสก์

เมื่อผู้วางระบบเลือกขนาดซับเน็ตแล้วจะกำหนดพารามิเตอร์เพื่อใช้บอกให้โฮสต์และเราเตอร์ทราบว่าซับเน็ตที่ใช้งานมีขนาดกี่บิต ค่านี้เรียกว่า ซับเน็ตมาสก์ (Subnet Mask)

ซับเน็ตมาสก์เป็นตัวเลข 32 บิต ซึ่งเขียนอยู่ในรูป Dotted-Decimal เช่นเดียวกับการเขียนไอพีแอดเดรส ซับเน็ตมาสก์จะมีบิตที่ตรงกับเลขเครือข่ายและเลขซับเน็ตเท่ากับ "1" ส่วนบิตที่ตรงกับเลยโฮสต์มีค่าเท่ากับ "0" การเลือกซับเน็ตมาสก์ควรใช้ค่าที่มีบิต "1" อยู่ติดกันจากทางซ้ายมือไปทางขวามือเสมอ

ตัวอย่างเครือข่าย 161.246.0.0 ซึ่งแบ่งให้มีเลขซับเน็ตและเลขโฮสต์อย่างละ 8 บิตจะมีค่าซับเน็ตมาสก์เท่ากับ 255.255.255.0 ค่านี้คำนวณได้จากการเขียนไอพีแอดเดรสทั้ง 4 หลัก และใส่เลขฐานสองค่า "1" ให้ครบทุกบิตที่เป็นเลขเครือข่ายและเลขซับเน็ต จากนั้นให้ใส่ค่า "0" สำหรับเลขโฮสต์ แล้วจึงแปลงเลขฐานสองที่

	8 บิต	8 บิต	8 บิต	8 บิต
1. นำค่าไอพีแอดเดรส	161	246	SubnetID	HostID
2. กำหนดบิต "1" และ "0"	11111111	11111111	11111111	00000000
3. แปลงเป็นเลขฐานสิบ	255	255	255	0

เครือข่าย 161.246.0.0 ซึ่งใช้ซับเน็ตมาสก์เท่ากับ 255.255.255.0 เรียกว่ามีซับเน็ตมาสก์ 24 บิต เนื่องจากมีบิตที่มีค่า "1" จำนวน 24 บิต หรือเขียนตามรูปแบบที่นิยมใช้ในปัจจุบันคือ 161.246.0.0/24 โดยเรียกว่าเครือข่าย 161.246.0.0 มีพรีฟิกซ์ 24 บิต

สังเกตว่า 161.246.0.0/24 ใช้เลขซับเน็ตจำนวน 8 บิต ดังนั้นนอกจากจะเรียกว่ามีพรีฟิกซ์ 24 บิตแล้ว ยังเรียกได้อีกว่าใช้ซับเน็ตบิตจำนวน 8 บิต

3.2.2 ดีฟอลต์ซับเน็ตมาสก์ (Default Subnet Mask)

การติดตั้งโฮสต์เข้าเครือข่ายนอกจากจะต้องกำหนดไอพีแอดเดรสแล้วต้องกำหนดค่าซับเน็ตมาสก์ตามที่ผู้ดูแลระบบกำหนดไว้ด้วย ถึงแม้ว่าในบางเครือข่ายเช่นเครือข่ายในคลาส C ซึ่งมีโฮสต์และไม่ได้แบ่งให้มีซับเน็ต ขั้นตอนการติดตั้งโฮสต์ยังจำเป็นต้องใส่ค่าซับเน็ตมาสก์เช่นกัน แต่ค่าซับเน็ตมาสก์นี้เรียกว่า ดีฟอลต์ซับเน็ตมาสก์ (Default Subnet Mask) ดีฟอลต์ซับเน็ตมาสก์ของเครือข่ายคลาส A, B และ C แสดงได้ดังตารางที่ 4.2

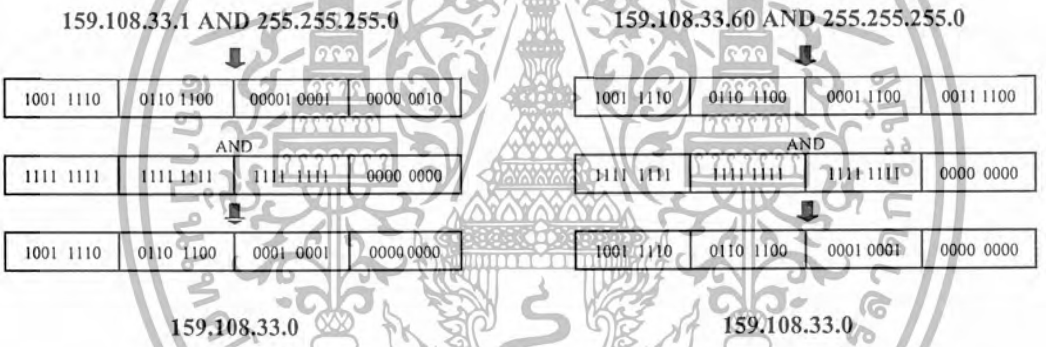
ผู้ดูแลระบบสามารถวางแผนจัดการเครือข่ายโดยเลือกทำซับเน็ตหรือไม่ทำซับเน็ตตามความต้องการ โดยปกติแล้วผู้ดูแลระบบเครือข่ายในคลาส A และ B ไม่สามารถหลีกเลี่ยงการใช้ซับเน็ตได้

คลาส	ดีฟอลต์ซับเน็ตมาสก์	ดีฟอลต์ซับเน็ตมาสก์(ฐาน 2)
A	255.0.0.0	11111111.00000000.00000000.00000000
B	255.255.0.0	11111111. 11111111.00000000.00000000
C	255.255.255.0	11111111.11111111.11111111.00000000

ตารางที่ 3-2 ค่าดีฟอลต์ซับเน็ตมาสก์

3.2.3 การเลือกเส้นทางในซับเน็ต

ซับเน็ตมาสก์นอกจากจะช่วยจัดแบ่งเครือข่ายย่อยแล้วยังใช้ประโยชน์ในการเลือกเส้นทางส่งไอพีดาต้าแกรมระหว่างเครือข่ายย่อยด้วย เช่น โฮสต์ 161.246.33.2 ในเครือข่าย 161.246.0.0/24 (ซับเน็ตมาสก์ 255.255.255.0) ต้องการส่งข้อมูลไปยังโฮสต์ 161.246.33.60 โพรโตคอลไอพีจะทำหน้าที่เลือกเส้นทางโดยนำแอดเดรส 161.246.33.2 และ 161.246.33.60 มาผ่านลอจิก "AND" บิตต่อบิตกับค่าซับเน็ตมาสก์ดังรูป



รูปที่ 3-6 การตรวจหาแอดเดรสซับเน็ตเพื่อเลือกเส้นทาง

ผลลัพธ์จากลอจิก "AND" ของแอดเดรสและเน็ตมาสก์ข้างต้นได้ค่าแอดเดรสซับเน็ต 161.246.33.0 เท่ากัน ซึ่งหมายความว่าโฮสต์ทั้งสองอยู่ในซับเน็ตเดียวกัน หากเครือข่ายที่ใช้คืออีเทอร์เน็ตแล้ว โฮสต์ 161.246.33.2 จะสร้างแพ็กเก็ตโดยระบุอีเทอร์เน็ตแอดเดรสของ 161.246.33.60 โดยไม่ต้องส่งแพ็กเก็ตให้เราเตอร์ดำเนินการ โปรดสังเกตว่าการใช้ลอจิก "AND" เป็นการใช้ซับเน็ตมาสก์เพื่อ "มาสก์" ให้ได้เฉพาะเลขเครือข่าย ค่าซับเน็ตมาสก์จึงเป็นเสมือน หน้ากาก ครอบเอาเลขเครือข่ายออกมา

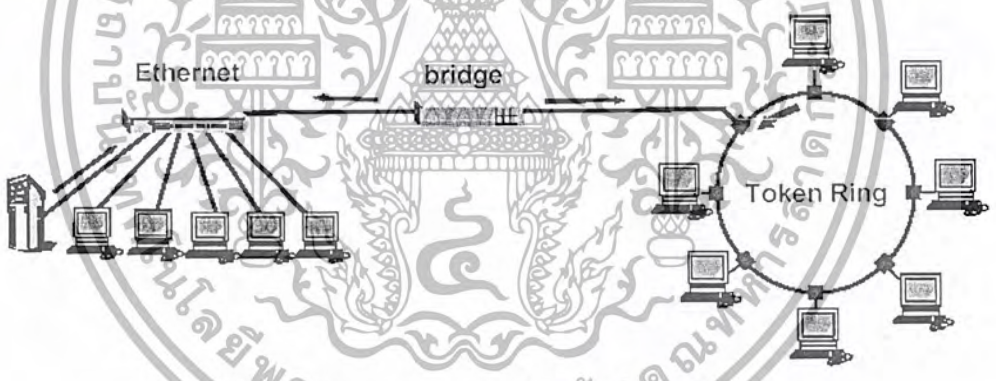
ในกรณีที่โฮสต์ปลายทางอยู่ต่างเครือข่ายกับโฮสต์ต้นทาง เช่น แอดเดรสของโฮสต์ต้นทางคือ 161.246.33.2 และโฮสต์ปลางทางคือ 161.246.40.5 ผลจากลอจิก "AND" ระหว่าง 161.246.40.5 กับมาสก์ 255.255.255.0 จะได้ค่า 158.0108.40.0 ซึ่งต่างจาก 161.246.33.0 ดังนั้นโฮสต์ 161.246.33.2 จะสรุปว่า 161.246.40.0 ซึ่งต่างจาก 161.246.33.0 ดังนั้นโฮสต์ 161.246.33.2 จะสรุปว่า 161.246.40.0 อยู่ต่างซับเน็ต และจะส่งแพ็กเก็ตไปยังเราเตอร์เพื่อให้เราเตอร์เพื่อให้เราเตอร์นำส่งแพ็กเก็ตต่อไป

บทที่ 4
พื้นฐานบริดจ์และสวิตช์

4.1 บริดจ์ (Bridge)

ในองค์กรขนาดใหญ่มักจะมีระบบเครือข่ายแลนอยู่หลายระบบ ซึ่งอาศัยการเชื่อมต่อโดยใช้เราเตอร์ (Router) สำหรับเครือข่ายที่ใช้โพรโตคอลอย่างเดียวกันเท่านั้น อุปกรณ์เครือข่ายประเภทบริดจ์ จึงได้รับการพัฒนาขึ้นมา เพื่อใช้ในการเชื่อมต่อระบบเครือข่ายต่างชนิดกันเข้าด้วยกัน โดยอาศัยแมคแอดเดรสในการกำหนดเส้นทางการสื่อสาร ดังนั้นบริดจ์จึงสามารถเรียกอีกอย่างได้ว่า Low-level router

เนื่องจากบริดจ์ทำงานในชั้นที่ 2 (Data Link Layer) ดังนั้นจึงมองไม่เห็นความแตกต่างของแพ็กเก็ต IP, IPX และอื่น ๆ ทำให้สามารถรับ-ส่งข้อมูลของโพรโตคอลได้เกือบทุกชนิด แต่การควบคุมเส้นทางในการส่งข้อมูลของบริดจ์ จะมีความยืดหยุ่นน้อยกว่าเราเตอร์ เนื่องจากจะใช้ข้อมูลแมคแอดเดรสเท่านั้นในการกำหนดเส้นทาง ดังนั้น บริดจ์จึงเหมาะสมกับระบบเครือข่ายที่มีความซับซ้อนไม่มากนัก



รูปที่ 4-1 แสดงระบบเครือข่ายที่ใช้บริดจ์ในการเชื่อมต่อ

4.1.1 ชนิดของบริดจ์

บริดจ์มีอยู่ 2 ชนิด คือ ทรานส์แพเร้นท์บริดจ์ (Transparent Bridge) และ ซอสเราท์บริดจ์ (Source Route Bridge)

4.1.1.1 ทรานส์แพเร้นท์บริดจ์ (Transparent Bridge)

พัฒนาโดย Digital Equipment Corporation ในต้นปี 1980 ซึ่งต่อมาได้มีการกำหนดมาตรฐาน IEEE 802.1 โดยผู้ใช้งานได้กำหนดความต้องการทรานส์แพเร้นท์บริดจ์ให้มีลักษณะดังนี้ แรกทีเดียวผู้ใช้ (ทุกระบบ) จะต้องไม่มีส่วนเกี่ยวข้องกับการทำงานของทรานส์แพเร้นท์บริดจ์ การมีอุปกรณ์ประเภทนี้อยู่ในระบบ หรือไม่

มีอยู่ก็ตาม จะต้องไม่มีผลกระทบใด ๆ ต่อผู้ใช้งาน ผู้ใช้ทั่วไปจะต้องสามารถซื้ออุปกรณ์นี้มาจากตัวแทนจำหน่ายของบริษัทใดก็ได้ การติดตั้งจะต้องมีความยุ่งยากเพียงแค่การเสียบปลั๊กไฟฟ้าและเสียบสายระบบเครือข่ายต่าง ๆ เข้ากับอุปกรณ์นี้ แม้ว่าจะไม่มีการกำหนดค่าพารามิเตอร์ใด ๆ ตัวอุปกรณ์ก็จะต้องสามารถทำงานได้ในทันที ผลที่เกิดขึ้นนั้นน่าแปลกใจเป็นอย่างยิ่งว่า ทรานส์แพเร้นท์บริดจ์นั้นมีตัวตนและทำงานได้อย่างที่ผู้ใช้ต้องการ

ทรานส์แพเร้นท์บริดจ์ ส่วนใหญ่จะใช้ในการเชื่อมต่อระหว่างเซกเมนต์ (Segment) ของอีเทอร์เน็ต โดยการทำงานของทรานส์แพเร้นท์บริดจ์ จะเก็บข้อมูลแมคแอดเดรสของสเตชันที่ต่ออยู่ของพอร์ตต่าง ๆ โดยจะใช้ข้อมูลนั้น เมื่อมีเฟรมส่งเข้ามาที่พอร์ตนั้น

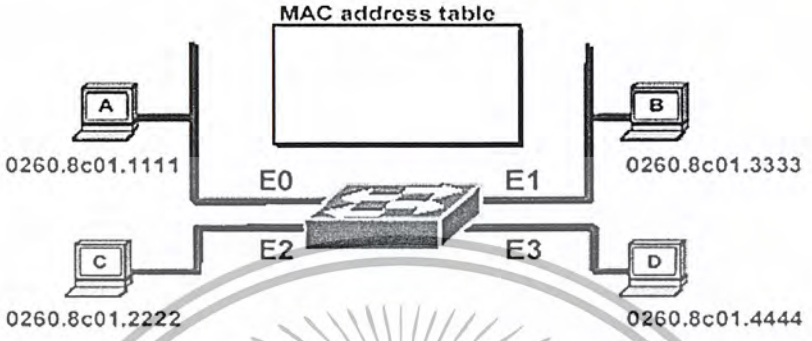
MAC Address Table
E0: 0260.8c01.1111
E2: 0260.8c01.2222
E1: 0260.8c01.3333
E3: 0260.8c01.4444

รูปที่ 4-2 แสดงตัวอย่าง MAC Address Table

เมื่อบริดจ์ได้รับเฟรมเข้ามา จะตรวจสอบแอดเดรสของสเตชันปลายทางและส่งเฟรมนั้น ออกไปยังพอร์ตที่ระบุ ยกเว้นเฟรมที่มีแอดเดรสของสเตชันปลายทางเป็นพอร์ทเดียวกันกับสเตชั่นต้นทาง แต่ละรายการในตารางจะมีเวลากำหนดไว้เรียกว่า Time-To-Live (TTL) โดยรายการนั้นจะถูกลบออกจากตารางเมื่อถึงเวลา TTL ที่กำหนดและ TTL จะถูกกำหนดใหม่เมื่อมีเฟรมจากสเตชันของรายการนั้นเข้ามาอีกครั้ง ซึ่งจะช่วยแก้ปัญหาในกรณีที่มีการย้ายสเตชันไปยังพอร์ตอื่น หรือการนำสเตชันนั้นออกจากระบบเครือข่าย ในกรณีที่บริดจ์ไม่สามารถหารายการที่ตรงกับแอดเดรสของสเตชั่นปลายทางได้ เฟรมนั้นจะถูกส่งไปยังทุกพอร์ตยกเว้นพอร์ตที่รับเฟรมเข้ามา

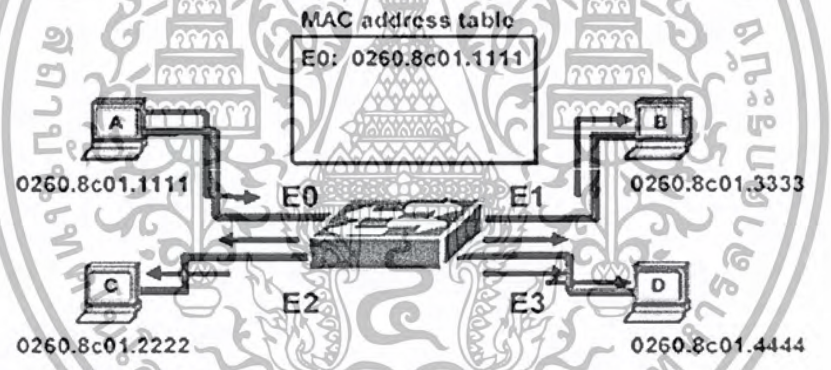
ทรานส์แพเร้นท์บริดจ์จะช่วยให้สามารถแบ่งระบบเครือข่ายออกเป็นเซกเมนต์ย่อย ๆ เพื่อลดปริมาณของการส่งข้อมูลในรูปแบบอีเทอร์เน็ตในอุปกรณ์ฮับไม่ให้คับคั่งมากเกินไป

ตัวอย่างการส่งข้อมูลและการสร้างตารางแมคแอดเดรส



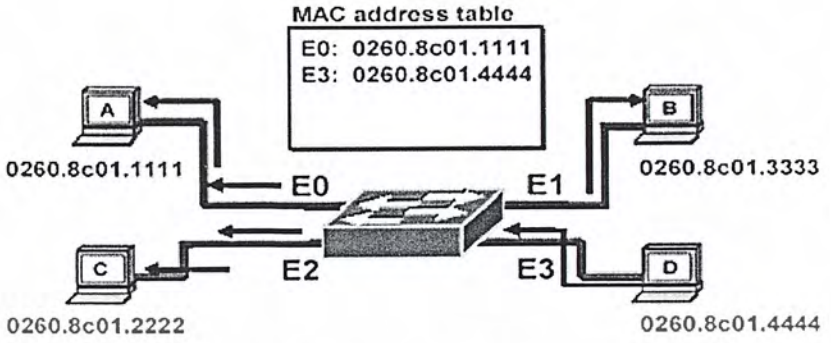
รูปที่ 4-3 แสดงการเรียนรู้แมคแอดเดรส

จากรูป เป็นตอนเริ่มต้น ยังไม่มีการส่งข้อมูล ดังนั้น ในตารางแมคแอดเดรสจึงยังว่างเปล่าอยู่ (empty)



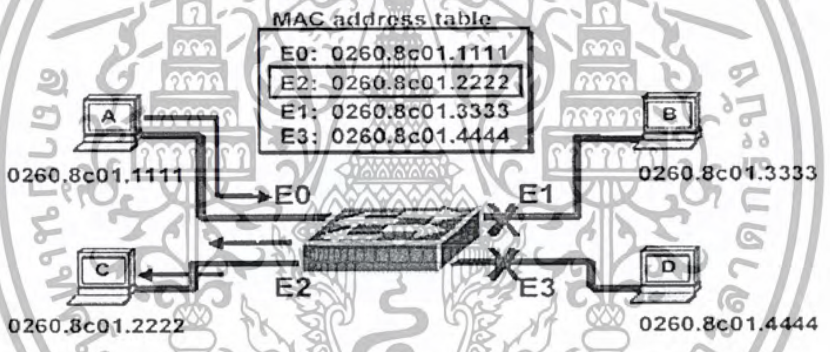
รูปที่ 4-4 แสดงการเรียนรู้แมคแอดเดรส

จากรูปสเตชัน A (Station A) ต้องการส่งข้อมูลไปยังสเตชัน C (Station C) ดังนั้นที่เฟรมข้อมูลของเอจะมีแอดเดรสของสเตชันต้นทาง (Source Address) เป็นแมคแอดเดรสของ A คือ 0260.8c01.1111 และแอดเดรสของสเตชันปลายทาง (Destination Address) เป็นแมคแอดเดรสของ C คือ 0260.8c01.2222 เมื่อสเตชัน A ส่งเฟรมไปยังสวิตช์ จะทำการดูที่สเตชันต้นทางทำการเรียนรู้ว่า พอร์ต E0 ที่รับข้อมูลเข้ามาคือ 0260.8c01.1111 จากนั้นทำการใส่ลงในตารางแมคแอดเดรสเป็น E0: 0260.8c01.1111 จากนั้นดูต่อไปที่สเตชันปลายทาง แล้วจึงไปค้นหาที่ตารางแมคแอดเดรสว่ามีแมคแอดเดรสนี้อยู่หรือไม่ จากรูป เราจะเห็นได้ว่า ในตารางไม่มีสเตชันปลายทางอยู่ เมื่อเป็นในกรณีนี้ สวิตช์จะทำการส่งออกไปยังทุกพอร์ต (Flood Out)



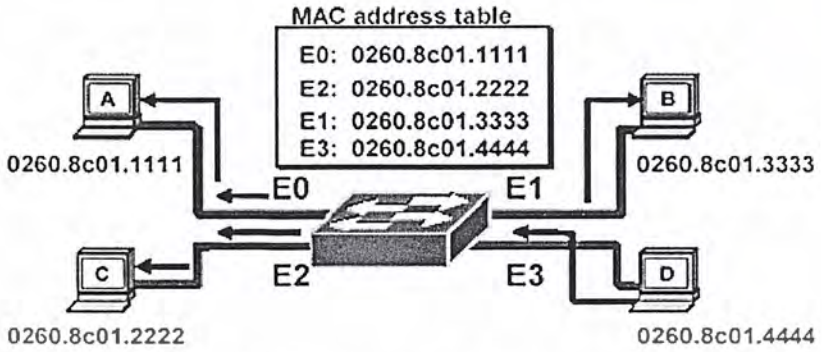
รูปที่ 4-5 แสดงการเรียนรู้แมคแอดเดรส

จากรูป สเตชัน D (Station D) ต้องการส่งเฟรมข้อมูลไปยังสเตชัน C (Station C) จะเป็นกรณีเดียวกับในรูป 4-4 นั่นเอง



รูปที่ 4-6 แสดงการทำงานของสวิตช์ในการทำฟิลเตอร์ (Filter)

จากรูปสเตชัน A ต้องการส่งเฟรมข้อมูลไปยังสเตชัน C เมื่อสวิตช์ทำการเรียนรู้แมคแอดเดรส แล้วจึงทำการค้นหาในตารางและพบว่าสเตชันปลายทางมีจุดหมายอยู่ที่พอร์ต E2 จึงทำการส่งเฟรมข้อมูลไปยังพอร์ต E2 เพียงพอร์ตเดียว ไม่ส่งไปยังพอร์ตอื่น ๆ



รูปที่ 4-7 แสดงการเรียนรู้แมคแอดเดรส

เมื่อสเตชัน D ต้องการส่งเฟรมข้อมูลแบบบอร์ดคาสต์ (Broadcast) หรือ เมาติคาสต์ (Multicast) สวิตช์ก็จะทำการส่งข้อมูลออกไปทุกพอร์ตยกเว้นพอร์ต E3 ซึ่งก็คือ พอร์ตที่รับเฟรมเข้ามานั่นเอง

4.1.1.2 ซอสเราท์บริดจ์ (Source-Route Bridge (SRB))

SRB เป็นอัลกอริธึมที่พัฒนาโดย IBM สำหรับการเชื่อมต่อระหว่างแลนแบบโทเก้นริง (IEEE 802.5) โดยการส่งข้อมูลแบบ SRB จะต้องมีการกำหนดเส้นทางก่อนล่วงหน้า ซึ่งมีขั้นตอนในการหาเส้นทาง คือ

เมื่อโฮสต์เอ็กซ์ (Host X) ต้องการส่งเฟรมให้โฮสต์วาย (Host Y) ในครั้งแรก โฮสต์เอ็กซ์จะไม่ทราบว่าโฮสต์วายอยู่ในเครือข่ายแลนเดียวกันหรือไม่ โฮสต์เอ็กซ์จะทำการส่งเฟรมทดสอบ ถ้าเฟรมกลับมายังโฮสต์เอ็กซ์ โดยที่บิต A ในเฟรมโทเก้นริงไม่เป็น 1 แสดงว่า โฮสต์วายอยู่ต่างเซกเมนต์

จากนั้น โฮสต์เอ็กซ์จะทำการส่งเฟรมเอกโพลเรอร์ (Explorer) ไปยังบริดจ์ บริดจ์เมื่อได้รับเฟรมเอกโพลเรอร์ จะส่งเฟรมนั้นไปยังทุกพอร์ตยกเว้นพอร์ตที่รับเฟรม โดยเพิ่มข้อมูลของเส้นทาง (Route) ในเฟรมเอกโพลเรอร์ตามบริดจ์นั้น เมื่อเฟรมเอกโพลเรอร์ไปถึงโฮสต์วายจะทำการตอบกลับมายังโฮสต์เอ็กซ์ตามข้อมูลเส้นทาง ซึ่งอาจจะตอบกลับมาหลายครั้งตามจำนวนเฟรมเอกโพลเรอร์ที่ได้รับ

ซึ่งโฮสต์เอ็กซ์จะต้องเลือกเส้นทางใดเส้นทางหนึ่ง โดยที่วิธีในการเลือกนั้นไม่ได้กำหนดไว้ใน IEEE 802.5 แต่สามารถเป็นไปได้หลายอย่าง เช่น

- เลือกเฟรมตอบกลับแรกที่ได้รับ
- เลือกเฟรมที่มีฮอป (Hop) น้อยที่สุด
- เลือกเส้นทางที่ยอมให้มีเฟรมขนาดใหญ่ที่สุด

เป็นต้น

เมื่อเลือกเส้นทางได้แล้ว ข้อมูลเส้นทางจะถูกกำหนดลงในเฟรมที่จะส่งไปที่โฮสต์วายในรูปแบบ Routing Information Field (RIF)

4.1.2 การเปรียบเทียบบริดจ์ในระบบ 802

บริดจ์ทั้งแบบทรานส์แพเร้นท์บริดจ์และแบบ SRB มีทั้งข้อดีและข้อเสียที่แตกต่างกันดังที่สรุปไว้ในตารางในรูปที่ 4-8

Issue	Transparent bridge	Source routing bridge
Orientation	Connectionless	Connection-oriented
Transparency	Fully transparent	Not transparent
Configuration	Automatic	Manual
Routing	Suboptimal	Optimal
Locating	Backward learning	Discovery frames
Failures	Handled by the bridges	Handled by the hosts
Complexity	In the bridges	In the hosts

ตารางที่ 4-1 ตารางเปรียบเทียบคุณสมบัติของทรานส์แพเร้นท์บริดจ์และแบบ SRB

หัวใจของความแตกต่างระหว่างบริดจ์ทั้ง 2 ชนิดคือ การสื่อสารเครือข่ายแบบมีการติดต่อช่วงสั้น (Connectionless) และแบบมีการติดต่ออย่างต่อเนื่อง (Connection-oriented) ทรานส์แพเร้นท์บริดจ์ไม่ใช้แนวคิดของวงจรเสมือน เส้นทางเดินของแต่ละเฟรมจะถูกเลือกอย่างเป็นอิสระ ส่วน SRB มีลักษณะตรงกันข้ามคือต้องมีการค้นหาเส้นทางเดินข้อมูลให้ได้เสียก่อน จากนั้นจึงใช้เส้นทางที่ค้นพบสำหรับการส่งข้อมูลจริงในภายหลัง

การทำงานของทรานส์แพเร้นท์บริดจ์จะไม่เข้าไปเกี่ยวข้องกับโฮสต์เลยแม้แต่น้อย และยังสามารถทำงานเข้ากันได้กับการส่งข้อมูลตามมาตรฐาน 802 ทุกระบบ ในขณะที่ SRB ต้องให้โฮสต์เข้ามาร่วมกระบวนการทำงานด้วยและไม่สามารถทำงานร่วมกับการส่งข้อมูลตามมาตรฐาน 802 บางระบบได้ นั่นคือโฮสต์จะต้องรู้จักโครงสร้างและการทำงานของบริดจ์เป็นอย่างดี และสามารถทำงานร่วมกันได้ การแบ่งระบบเครือข่ายออกเป็น 2 วงที่เชื่อมกันโดยวิธีการเลือกทางเดินโดยผู้ส่งข้อมูลจะต้องทำการเปลี่ยนแปลงโปรแกรมของโฮสต์ด้วย

การใช้ทรานส์แพเร้นท์บริดจ์ไม่จำเป็นต้องมีระบบบริหารเครือข่าย บริดจ์สามารถเรียนรู้ได้ด้วยตัวเองและสามารถปรับตัวให้เข้ากับระบบเครือข่ายนั้น ๆ ได้โดยอัตโนมัติ ส่วน SRB จะต้องให้ผู้บริหารเครือข่ายทำการติดตั้งหมายเลขระบบเครือข่ายและหมายเขบริดจ์ด้วยตนเองทั้งหมด ความผิดพลาดเช่น ระบบเครือข่ายหรือบริดจ์ใช้หมายเลขซ้ำกันนั้นตรวจสอบได้ยากมาก ซึ่งอาจจะทำให้เกิดการรวบซ้ำของเฟรมข้อมูลได้ นอกจากนี้การติดต่อของระบบเครือข่าย 2 แห่งที่เคยเชื่อมต่อกันในอดีตนั้น สำหรับทรานส์แพเร้นท์บริดจ์แล้วไม่มีสิ่งใดต้องทำยกเว้นการเชื่อมต่อสายเข้าด้วยกันเท่านั้น ส่วนการใช้ SRB อาจมีความจำเป็นจะต้องเปลี่ยนหมายเลขของระบบเครือข่ายหลาย ๆ ระบบที่ใช้หมายเลขซ้ำกัน

ข้อดีประการหนึ่งของ SRB อยู่ที่ระบบนี้สามารถใช้เส้นทางการส่งข้อมูลที่ดีที่สุดได้ (ในทางทฤษฎี) ในขณะที่ทรานส์แพเร้นท์บริดจ์มีข้อจำกัดที่เกิดขึ้นจากการใช้สแปนนิ่งทรี นอกจากนี้ SRB สามารถเลือกใช้งานบริดจ์คู่ขนานระหว่างระบบเครือข่ายได้อย่างเหมาะสม แต่บริดจ์ที่ใช้งานจริงจะฉลาดมากพอที่จะแบ่งงานกันทำให้ได้ตามที่กล่าวไว้หรือไม่นั้น ยังไม่มีการพิสูจน์

การค้นหาตำแหน่งของผู้รับโดยวิธีการเรียนรู้ย้อนหลังในทรานส์แพเร้นท์บริดจ์มีข้อจำกัดตรงที่บริดจ์จะต้องรอจนกระทั่งเฟรมที่ส่งมาจากสถานีต่าง ๆ นั้นมาถึงจึงจะสามารถเรียนรู้จากข้อมูลเหล่านั้นได้ ส่วนการค้นหาเฟรมใน SRB มีปัญหาเกี่ยวกับการเพิ่มจำนวนของเฟรมค้นหาอย่างรวดเร็วมาก โดยเฉพาะในระบบที่มีจำนวนระบบเครือข่ายมากและใช้บริดจ์คู่เชื่อมต่อระหว่างเครือข่ายเข้าด้วยกัน

การจัดการความผิดพลาดของบริดจ์ทั้งสองแบบมีวิธีการที่แตกต่างกัน ในแบบแรกบริดจ์สามารถเรียนรู้เกี่ยวกับบริดจ์และระบบเครือข่ายต่าง ๆ ที่ทำงานผิดพลาดหรือการเปลี่ยนแปลงรูปแบบเครือข่ายได้อย่างรวดเร็วและเป็นไปอย่างอัตโนมัติ ด้วยการดักฟังสัญญาณที่ส่งออกมาจากอุปกรณ์เหล่านั้นเพียงอย่างเดียว โฮสต์จะไม่ต้องเข้ามายุ่งเกี่ยวด้วยเลย

ส่วนการจัดการความผิดพลาดในระบบของ SRB มีวิธีการที่แตกต่างออกไปอย่างสิ้นเชิง เมื่อบริดจ์หยุดทำงาน สเตชันที่เลือกเส้นทางเดินข้อมูลผ่านอุปกรณ์ตัวนั้นจะพบว่าเฟรมที่ส่งออกไปไม่ได้รับการตอบรับกลับมาเลย สเตชันนั้นอาจส่งข้อมูลซ้ำแล้วซ้ำอีก สุดท้ายสเตชันนั้นจะทราบว่าเกิดมีอุปกรณ์บางอย่างทำงานผิดปกติ แต่ก็ยังไม่ทราบว่ามีปัญหาเกิดขึ้นที่สเตชันปลายทางหรืออยู่ในเส้นทางปัจจุบัน การหาคำตอบด้วการส่งเฟรมค้นหาข้อมูลออกไปใหม่จะทำให้ทราบว่าสเตชันปลายทางยังคงทำงานอยู่หรือไม่ อย่างไรก็ตาม ในกรณีที่บริดจ์หลักเสียหายหรือหยุดทำงานจะทำให้โฮสต์จำนวนมากเสียเวลาไปกับการรอคอยและส่งเฟรมค้นหาออกไปจนกว่าปัญหานั้นจะได้รับการแก้ไขแม้ว่าจะมีเส้นทางอื่นอยู่ก็ตาม การชำรุดของอุปกรณ์เป็นจุดอ่อนหลักของการเชื่อมต่อแบบต่อเนื่องทั้งหมด

4.2 สวิตช์ (Switch)

สวิตช์เป็นอุปกรณ์ในระบบเครือข่ายที่ออกแบบมาเพื่อแยกระบบเครือข่ายออกเป็นส่วนย่อย ๆ เพื่อเพิ่มประสิทธิภาพของระบบเครือข่ายและทำให้การควบคุมระบบเครือข่ายทำได้ดีขึ้น โดยแต่ละพอร์ตของสวิตช์จะเป็นเซกเมนต์หนึ่งของระบบเครือข่าย ข้อมูลที่ส่งในเซกเมนต์เดียวกันจะไม่ถูกส่งไปยังเซกเมนต์อื่น เป็นการช่วยลดปัญหาความคับคั่งของข้อมูลได้

สวิตช์จะมีลักษณะคล้ายกับบริดจ์ในการแบ่งระบบเครือข่ายออกเป็นส่วนย่อย ๆ ในกรณีที่มีการส่งข้อมูลข้ามเซกเมนต์ สวิตช์จะส่งเฟรมไปยังพอร์ตที่สเตชันปลายทางอยู่เท่านั้น อีกทั้งสวิตช์ยังสามารถส่งข้อมูลระหว่างเซกเมนต์ได้พร้อม ๆ กันโดยไม่เกิดปัญหาการชนกันของข้อมูล (Collision) และสามารถส่งข้อมูลได้ในแบบสองทาง (Full-duplex)

สวิตช์จะทำงานที่ชั้น 2 (Data link Layer) ของ OSI Reference Model (โดยในปัจจุบัน สวิตช์สามารถทำงานได้ที่ชั้นที่ 2 ชั้นที่ 3 และชั้นที่ 4 แล้ว) ดังนั้นสวิตช์จะรับและส่งเฟรมข้อมูลตาม MAC Address ของสเตชันที่ต่ออยู่ที่พอร์ตของสวิตช์ โดยการต่อเชื่อมกับสวิตช์แบ่งออกเป็น 2 แบบคือ Segment switch และ Port switch

Segment Switch จะรองรับทราฟฟิกของสเตชันในเซกเมนต์ในแต่ละพอร์ต รวมทั้งเซกเมนต์ที่มีสเตชันเดียวด้วย ซึ่งจะเชื่อมต่อจากสเตชันมาที่พอร์ตของสวิตช์โดยตรง ซึ่งทำให้ผู้ออกแบบระบบเครือข่ายสามารถจัดให้สเตชั่นที่ต้องมีการส่งข้อมูลกันมาก ๆ หรือบ่อย ๆ อยู่ในเซกเมนต์เดียวกัน และสามารถจัดให้เซิร์ฟเวอร์ที่ให้บริการ

Port Switch หรือเรียกอีกอย่างหนึ่งว่า Switch Hub เป็นการใช้งานในลักษณะ 1 พอร์ตต่อ 1 สเตชัน โดยใช้งานแทนที่ฮับ

4.2.1 คัททรูสวิตช์ (Cut-Through Switching)

โดยการทำงานปกติของสวิตช์ จะทำการรับเฟรมเข้ามาก่อน แล้วจึงส่งเฟรมนั้นไปยังพอร์ตของสเตชันปลายทาง (Store & Forward) สวิตช์แบบคัททรูจะลดการหน่วงเวลาในขั้นตอนนี้ โดยเมื่อสวิตช์ได้รับข้อมูลเฟรมเพียงพอที่จะกำหนดสเตชั่นเป้าหมายได้แล้ว ก็จะเริ่มต้นการส่งข้อมูลทันทีโดยไม่ต้องรอให้ได้รับเฟรมทั้งหมด

การใช้งานสวิตช์แบบคัททรูอาจจะทำให้เกิดปัญหาการส่งเฟรมที่มีข้อผิดพลาดได้ ดังนั้นจึงควรกำหนดให้สวิตช์ทำการรับเฟรมมาจำนวนหนึ่งก่อน แล้วจึงเริ่มการส่งเฟรม เพื่อให้แน่ใจว่า เฟรมนั้นเป็นเฟรมที่ไม่มีข้อผิดพลาด

4.2.2 ชนิดของสวิตช์

Crossbar Switch เป็นสวิตช์ที่พัฒนาขึ้นในยุคแรก ๆ โดยทุกอินพุตจะต่อเข้ากับทุก ๆ เอาต์พุต โดยจะมีบัฟเฟอร์ของอินพุตที่ใช้ในการพักข้อมูลเมื่อพอร์ตเอาต์พุตกำลังใช้งานอยู่

Shared-memory Switch สวิตช์ชนิดนี้จะเก็บข้อมูลที่รับเข้าไว้ในหน่วยความจำและส่งออกไปยังพอร์ตของสเตชั่นปลายทาง ข้อมูลจะเข้าและออกระหว่างพอร์ตกับหน่วยความจำโดยตรง วิธีการนี้มีข้อเสียคือ เกิดความล่าช้าในการเก็บข้อมูลลงในหน่วยความจำ

High-speed bus Switch ข้อมูลที่เข้ามาที่พอร์ตจะส่งผ่านบัสและส่งออกไปยังพอร์ตที่สเตชั่นปลายทางเชื่อมต่ออยู่ บัสที่ใช้จะเป็นบัสความเร็วสูง โดยใช้เทคนิค TDM ในการให้บริการกับพอร์ตต่าง ๆ ซึ่งจะต้องมีบัฟเฟอร์ที่ใช้ในการพักข้อมูลไว้ชั่วคราว

สวิตช์แบบ High-speed bus เป็นชนิดที่มีการนำมาใช้มากที่สุด เช่น สวิตช์รุ่น Catalyst 3000 ของซิสโก้ (Cisco) ที่มีพอร์ต 10Base-T 16 พอร์ตและรองรับการเชื่อมต่อแบบฟาสต์อีเทอร์เน็ต (Fast Ethernet), ATM หรือแวน (WAN) จะใช้บัสความเร็ว 480 Mbps และมีบัฟเฟอร์ขนาด 256 K โดยใช้ชิปโพรเซสเซอร์ Intel i960 ในการควบคุมการเข้าถึงบัสของแต่ละพอร์ต

4.2.3 สวิตช์เลเยอร์ที่ 3 (Layer 3 Switch)

สวิตช์เลเยอร์ที่ 3 หรือ L3 Switch คือสวิตช์ที่ทำงานในระดับชั้นที่ 3 (Network Layer) ดังนั้นการเลือกเส้นทางการส่งข้อมูลของสวิตช์ L3 จึงต้องอาศัยข้อมูลที่อยู่ในแพ็กเก็ตของชั้นที่ 3 เช่นเดียวกับเราเตอร์ นอกจากนี้ยังต้องทำหน้าที่อื่น ๆ ที่กำหนดในชั้นที่ 3 ด้วย เช่น การตรวจสอบความถูกต้องของข้อมูลโดยการ checksum การตรวจสอบการหมดอายุของแพ็กเก็ต (TTL) การรองรับโพรโตคอลการจัดการต่าง ๆ ของชั้นที่ 3 และระบบควบคุมการปลอดภัย

Characteristic	Layer 3 Switch	Router
LAN Protocol (IP, IPX, Apple Talk)	Yes	Yes
Subnet definition	Layer 2 Switch domain	Port
Forwarding architecture	Hardware	Software (ASIC)
Management	SNMP, RMON	SNMP (RMON)
WAN support	No	Yes
Price	Low	High

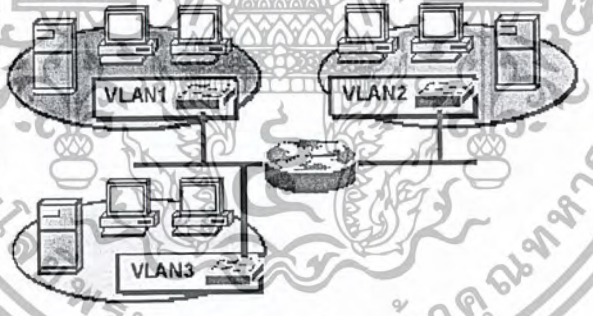
ASIC – Application Specific Integrated Circuit

ตารางที่ 4-2 แสดงตารางการเปรียบเทียบระหว่างสวิตช์ชั้นที่ 2 กับสวิตช์ชั้นที่ 3

บทที่ 5
แลนเสมือน

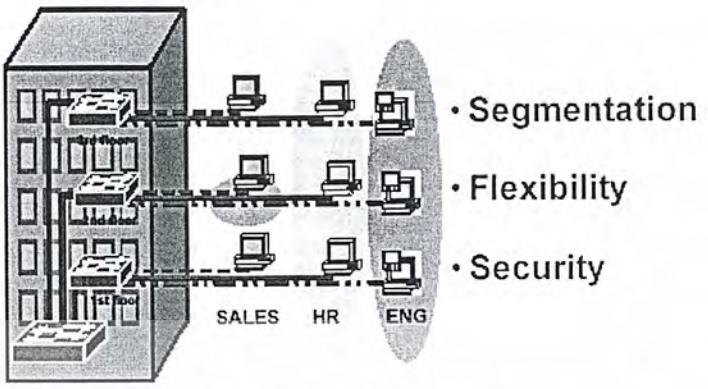
เมื่อพิจารณาถึงระบบเครือข่ายที่ประกอบด้วยอุปกรณ์ในชั้นที่ 2 เท่านั้น เช่น เซกเมนต์ของอีเทอร์เน็ต, สวิตช์ที่มีหลายพอร์ต หรือเครือข่ายที่ประกอบไปด้วยสวิตช์หลาย ๆ ตัว เครือข่ายแบบนี้ เรียกว่า Flat Network Topology เครือข่ายแบบนี้จะมีได้เพียง 1 บอร์ดคาสต์โดเมน (Broadcast Domain) เท่านั้น หมายความว่า เมื่อมีการส่งเฟรมแบบบอร์ดคาสต์จะทำให้ทุก ๆ สเตชันได้รับเฟรมนี้ไป ดังนั้นยิ่งมีจำนวนของอุปกรณ์ (เช่น สวิตช์, ฮับ, สเตชัน) มากขึ้นเท่าใด ก็ยิ่งทำให้เกิดทราฟฟิกขึ้นในเครือข่ายมากขึ้นเท่านั้น จนอาจทำให้เกิดเป็นบอร์ดคาสต์สตอร์ม (Broadcast Storm) ได้

แลนเสมือน หรือวีแลน (Virtual LANs (VLANs)) คือการสร้างเซกเมนต์ของระบบเครือข่ายที่ไม่ขึ้นกับระบบเครือข่ายทางกายภาพ หมายความว่าเราสามารถแบ่งเครือข่ายเราออกเป็นเครือข่ายย่อย ๆ ได้ โดยไม่ขึ้นต่อกัน เมื่อกำหนดวีแลนขึ้นมาแล้ว เราจะถือว่า แต่ละวีแลนเป็น 1 บอร์ดคาสต์โดเมนเป็นเครือข่ายแลนที่ไม่เกี่ยวข้องต่อกัน



รูปที่ 5-1 แสดงเครือข่ายวีแลน

เมื่อกำหนดวีแลนแล้ว สเตชันในวีแลนเดียวกันจะสามารถส่งข้อมูลถึงกันได้ แต่ถ้าเป็นการส่งข้อมูลข้ามเซกเมนต์จะต้องใช้เราเตอร์ในการส่งผ่านข้อมูล ซึ่งผู้ดูแลระบบสามารถกำหนดรายละเอียดของการส่งผ่านข้อมูลระหว่างเซกเมนต์ได้



รูปที่ 5-2 แสดงตัวอย่างการแบ่งวีแลนออกเป็นแผนกงาน

วีแลนมีข้อดีคือ

1.*Segmentation* คือสามารถแบ่งเครือข่ายออกเป็นเครือข่ายย่อยได้ เป็นการแบ่งทราฟฟิกของแต่ละวีแลนออกจากัน

2.*Flexibility* คือ มีความยืดหยุ่นในการเปลี่ยนแปลงสมาชิกที่อยู่ในวีแลนได้ง่าย

3.*Security* คือ เมื่อเราทำการแบ่งวีแลนแล้ว จะถือว่าแต่ละวีแลนเป็น 1 บอร์ดคาสต์โดเมนทำให้การส่งข้อมูลไม่รั่วไหลออกไปยังวีแลนอื่น ๆ เป็นข้อมูลที่ส่งอยู่ในวีแลนเดียวเท่านั้น

5.1 ประเภทของวีแลน

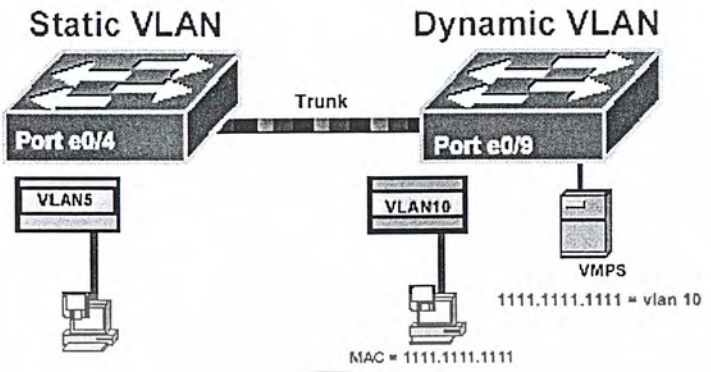
แบ่งออกเป็น 2 ประเภทใหญ่ ๆ คือ

5.1.1 สแตติกวีแลน (Static VLANs)

เป็นการกำหนดวีแลนจากพอร์ตของสวิตช์ว่า ต้องการให้พอร์ตไหนเป็นวีแลนใด เมื่อเรานำอุปกรณ์ไปต่อ ก็จะทำให้อุปกรณ์ชิ้นนั้นเป็นสมาชิกของวีแลนนั้นโดยอัตโนมัติ มีข้อดีคือ กำหนดได้ง่าย และดูแลง่าย (Based on port)

5.1.2 ไดนามิกวีแลน (Dynamic VLANs)

เป็นการกำหนดวีแลนตามค่าแมคแอดเดรสที่ได้กำหนดไว้ โดยจะมีฐานข้อมูล (Database) เก็บไว้ว่าแมคแอดเดรสค่าใดเป็นสมาชิกของวีแลนใด จะมีข้อดีเมื่อเราทำการเคลื่อนย้ายอุปกรณ์ใด ๆ ก็จะยังทำให้อุปกรณ์ตัวนั้นเป็นสมาชิกของวีแลนเดิมอยู่โดยอัตโนมัติ ไม่จำเป็นต้องไปกำหนดค่าใหม่ (Based on MAC Address)



รูปที่ 5-3 แสดงประเภทของวีแลน

นอกจากนี้ เรายังสามารถแบ่งประเภทวีแลนออกได้เป็นอีกหลายแบบ คือ

1. **Port-Based & MAC-Based**

   Port-Based : กำหนดวีแลนตามพอร์ตที่กำหนดไว้ (เหมือน Static VLAN)

   MAC-Based : กำหนดวีแลนตามแมคแอดเดรสที่กำหนดไว้ (เหมือน Dynamic VLAN)

2. **Protocol-Based & Dynamic-Based**

   Protocol-Based : กำหนด วีแลน ตามโพรโตคอลที่กำหนดไว้ เช่น

   Host X ใช้โพรโตคอล IP ดังนั้น จะเป็นสมาชิกของวีแลน 1

   Host Y ใช้โพรโตคอล IPX ดังนั้น จะเป็นสมาชิกของวีแลน 2

   Dynamic-Based : กำหนดวีแลนตาม User Profile ที่กำหนดไว้ โดยเก็บ User Profile ไว้ในฐานข้อมูล เช่น โฮสต์ X ทำการ Log in ตัวโพรไฟล์ (Profile) ของ โฮสต์ X จะเป็นตัวกำหนดให้โฮสต์ X เป็นของ วีแลน 1

5.2 ประเภทของการเชื่อมต่อ

ในการกำหนดวีแลนนั้น บางครั้งอาจมีการกำหนดให้ในวีแลนเดียวกันมีอุปกรณ์ที่เป็นสมาชิกอยู่ในสวิตช์คนละตัวกัน ดังนั้นจึงต้องมีการกำหนดการเชื่อมต่อของวีแลนเพื่อใช้เป็นกฎในการส่งข้อมูลภายในวีแลนเดียวกัน

5.2.1 **แอ็กเซสลิงค์ (Access Link)** เป็นการเชื่อมต่อที่บอกว่าลิงค์ (Link) นี้เป็นวีแลนใด โดยข้อมูลที่ผ่านจะมีแค่วีแลนเดียว

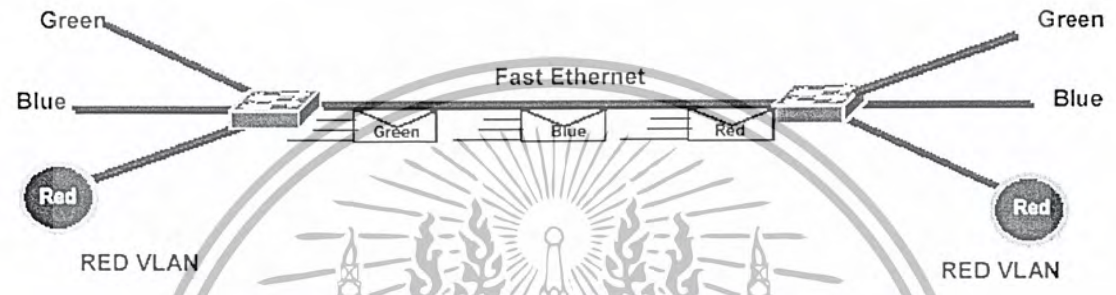
5.2.2 **ทรังค์ลิงค์ (Trunk Link)** เป็นการเชื่อมต่อที่ใช้ในการส่งข้อมูลได้หลายๆ วีแลน

รูปที่ 5-4 แสดงตัวอย่างของทรังค์ลิงค์

5.3 วิธีการระบุถึงวีแลน

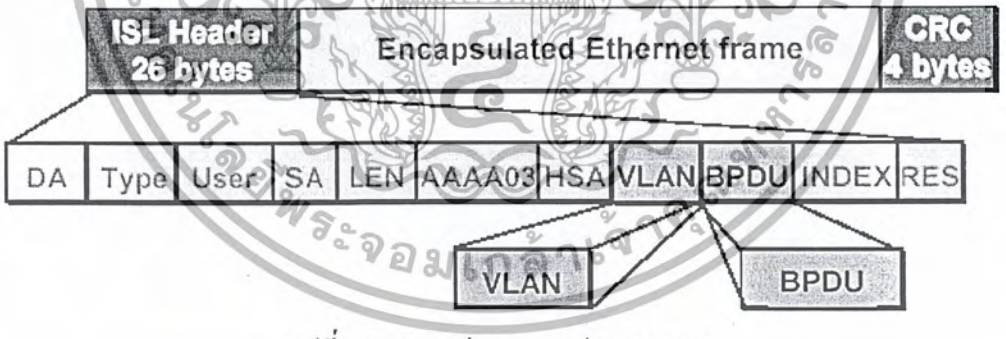
แพ็กเก็ตจะถูกส่งไปตาม Trunk Link โดยบรรจุข้อมูลที่ระบุถึงวีแลนไว้ในส่วนของเฮดเดอร์ (Header) ของแพ็กเก็ต ซึ่งวิธีการระบุนั้นมีอยู่ 2 แบบคือ

- Cisco ISL
- IEEE 802.1Q



รูปที่ 5-5 แสดงเครือข่ายของวีแลนที่ใช้การเชื่อมต่อแบบทรังค์ลิงค์

Cisco ISL



รูปที่ 5-6 แสดงรูปแบบของเฟรมของ ISL

เมื่อได้รับเฟรมอีเทอร์เน็ตมาแล้ว จะทำการ encapsulate ด้วย ISL Header และ CRC ซึ่งสนับสนุนวีแลนได้มากสุดถึง 1024 วีแลน

IEEE 802.1Q

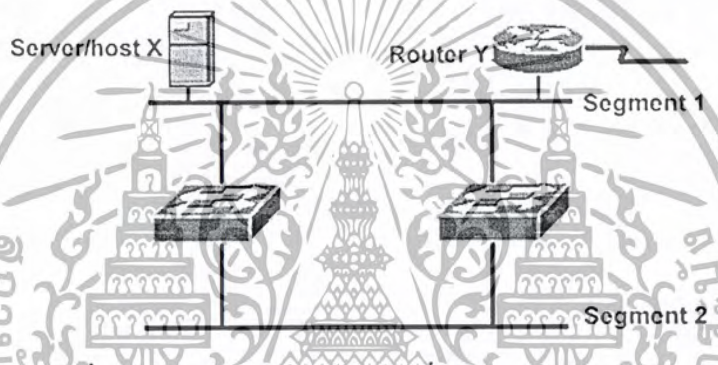
Initial MAC Address	2-Byte TPID 2-Byte TCI	Initial Type/Data	New CRC

รูปที่ 5-7 แสดงรูปแบบของเฟรมของ IEEE 802.1Q

- 2-byte tag protocol identifier (TPID)
- 2-byte tag control information (TCI)

บทที่ 6
สแปนนิ่งทรีโพรโตคอล

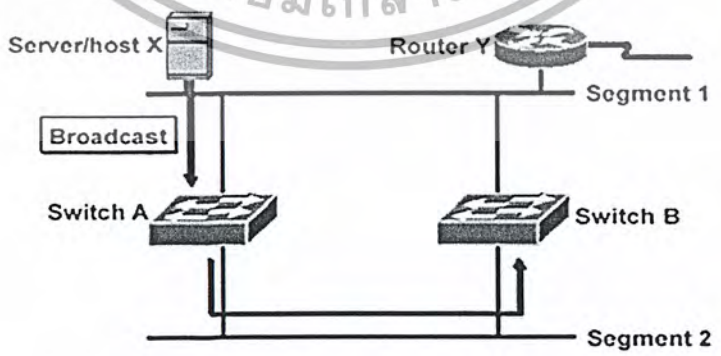
ในการสร้างระบบเครือข่ายที่ดีนั้น ควรต้องคำนึงถึงความน่าจะเป็นในทุกด้าน เมื่อเกิดกรณีฉุกเฉิน ระบบเครือข่ายจะทำการแก้ไขเช่นไร เช่น ถ้าในระบบ สวิตช์เกิดเสีย ใช้งานไม่ได้ จะทำเช่นไร ด้วยเหตุนี้ จึงทำให้เกิดเป็นแนวคิด Redundant Topology ขึ้น คือเป็นการสร้างระบบเครือข่ายแบบต่อให้มี สวิตช์ 2 ตัวในเส้นทาง (path) เดียวกัน เพื่อป้องกันในเหตุที่ถ้ามีสวิตช์ตัวใดตัวหนึ่งเสีย สวิตช์ตัวที่เหลือจะยังสามารถใช้งานต่อไปได้ ทำให้ระบบเครือข่ายไม่หยุดชะงัก



รูปที่ 6-1 แสดงตัวอย่างของเครือข่ายที่ใช้ Redundant Topology

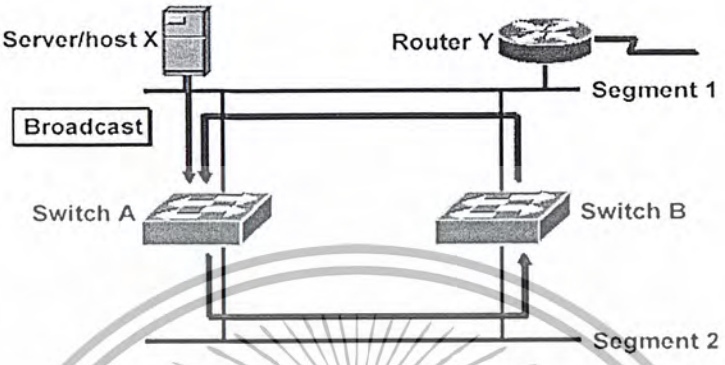
เมื่อเราสร้างระบบเครือข่ายโดยใช้ Redundant Topology แล้ว อาจทำให้เกิดลูป (loop) ขึ้นได้ ซึ่งเป็นสาเหตุทำให้เกิดบอร์ดคาสต์สตอร์มขึ้นในระบบเครือข่ายจนอาจเป็นสาเหตุทำให้เครือข่ายล่มได้

โดยขั้นตอนของการเกิดลูปจะเป็นดังนี้คือ



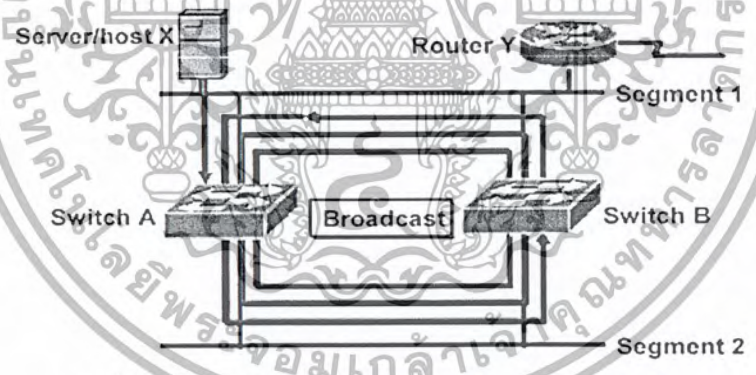
รูปที่ 6-2 แสดงการส่งข้อมูลจากโฮสต์ X ไปยังสวิตช์ A

เมื่อโฮสต์ X ส่งเฟรมข้อมูลออกไปให้กับสวิตช์ A จากนั้นสวิตช์ A จะทำการใส่แอดเดรสต้นทาง (Source Address) เข้าไปในตารางแมคแอดเดรสแล้วจึงทำการส่งออกไปยังทุกพอร์ตออกไป ทำให้สวิตช์ B ได้รับเฟรมข้อมูลนี้ด้วย



รูปที่ 6-3 แสดงการฟลัดเอาท์ (Flood out) ของสวิตช์ A

เมื่อสวิตช์ B ได้รับเฟรมข้อมูล ก็จะทำการใส่แอดเดรสต้นทางลงไปในตารางแมคแอดเดรสแล้วทำการส่งออกไปยังทุกพอร์ต (Flood Out) อีกครั้ง ทำให้สวิตช์ A ได้รับเฟรมข้อมูลนี้อีกครั้ง



รูปที่ 6-4 แสดงการเกิดบอร์ดคาสต์สตอร์ม (Broadcast Storm)

เมื่อสวิตช์ A ได้รับเฟรมข้อมูล ก็จะกระทำการเหมือนเดิม และต่อเนื่องไปถึงสวิตช์ B วนเป็นลูปไปเรื่อย ๆ ไม่รู้จบ และแม้ว่า เฟรมข้อมูลไปถึง ณ จุดหมายที่สเตชันที่ต้องการแล้ว ตัวสวิตช์ A และสวิตช์ B ก็จะยังส่งเฟรมข้อมูลกันต่อไปแบบไม่รู้จบ จนทำให้เกิดเป็นบอร์ดคาสต์สตอร์ม สร้างทราฟฟิกให้กับระบบเครือข่าย จนอาจทำให้เครือข่ายหยุดชะงักได้

6.1 สแปนนิ่งทรีโพรโตคอล (Spanning-Tree Protocol)

สแปนนิ่งโพรโตคอลถูกสร้างมาเพื่อใช้ในการกำจัดลูปที่เกิดขึ้นในระบบเครือข่าย โดยจะเสมือนจัดให้ระบบเครือข่ายมีลักษณะเหมือนกับต้นไม้ (Tree) คือจากจุดเริ่มต้นแล้วมีการแตกกิ่งก้านออกมาเรื่อย ๆ เป็นสายแห่งการติดต่อสื่อสาร จะทำให้เป็นเครือข่ายที่ไม่มีลูปเลย

หลักการทำงานของสแปนนิ่งโพรโตคอลคือ สวิตช์ทุกตัวจะทำการส่งเฟรมชื่อว่า Bridge Protocol Data Unit (BPDU) ออกไปยังทุกพอร์ต เพื่อแสดงถึงความมีตัวตนของสวิตช์ แล้วสวิตช์ทุกตัวจะได้รับ BPDU จากสวิตช์ตัวข้างเคียงเพื่อนำมาใช้ในการคำนวณโดยใช้สแปนนิ่งอัลกอริธึม (Spanning Tree Algorithm) ซึ่งมีหลักการคือ

- เลือกรูทบริดจ์ (Root Bridge)
- เลือกรูทพอร์ต (Root Port)
- เลือกดีไซเนเตทพอร์ต (Designated Port)

สวิตช์ทุกตัวจะทำการส่ง BPDU ออกไปยังทุกพอร์ต เป็นการแสดงถึงความมีตัวตนของสวิตช์ จากนั้นสวิตช์ทุกตัวจึงจะได้รับ BPDU มาเพื่อทำการคำนวณโดยใช้สแปนนิ่งอัลกอริธึมจะได้เป็นสแปนนิ่งทรีโพรเซส (Spanning-Tree Process) ซึ่งก็คือ

6.1.1 เลือกรูทบริดจ์

Root Bridge คือจุดที่เป็นจุดอ้างอิง (Reference) ของสแปนนิ่งทรี ซึ่งก็คือ จุดยอดของต้นไม้นั่นเอง โดยรูทบริดจ์จะเป็นสวิตช์ตัวที่มีบริดจ์ไอดี (Bridge ID) น้อยที่สุด

บริดจ์ไอดีประกอบด้วย

- Bridge Priority (2 bytes) ไพรออริตี้ (Priority) หรือ Weight ของสวิตช์สามารถมีแค่ได้ตั้งแต่ 0 – 65535 และมีค่าดีฟอลต์ (default) คือ 32768
- แมคแอดเดรส (8 bytes) เป็นสิ่งที่แสดงในเห็นถึงความเป็นเอกของแต่ละสวิตช์ เนื่องจากแมคแอดเดรสมีคุณสมบัติคือ มีเพียงแมคแอดเดรสเดียวในโลก ไม่เหมือนใคร และไม่สามารถเปลี่ยนแปลงได้

ในตอนเริ่มต้นนั้น สวิตช์จะตั้งค่ารูทบริดจ์เป็นบริดจ์ไอดีตัวเองก่อน จากนั้นจึงมีการรับ BPDU จากสวิตช์ข้างเคียง แล้วนำมาคำนวณหาตัวที่น้อยกว่า ทำเช่นนี้ไปเรื่อย ๆ จนกว่า สวิตช์ทุกตัวจะมี Root Bridge เป็นตัวเดียวกัน และหลังจากนั้นสวิตช์ก็ยังจะส่ง BPDU ทุก ๆ 2 วินาที (เป็นค่าดีฟอลต์)

6.1.2 เลือกรูทพอร์ต

สำหรับทุก ๆ นอนรูทบริดจ์ (Non-root Bridge) คือสวิตช์ที่ไม่ใช่รูทบริดจ์ต้องทำการเลือกรูทพอร์ต โดยเลือกพอร์ตที่ดีที่สุดในการสื่อสารไปยังรูทบริดจ์ โดยคำนวณจากพอร์ตคอสท์ (Port cost) หรือรูทพาร์ธคอสท์ (Root path cost – จำนวนฮอปทั้งหมดจากรูทบริดจ์จนถึงสวิตช์)

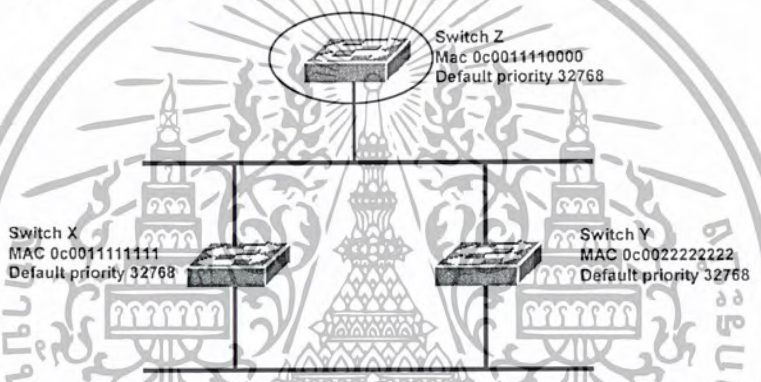
6.1.3 เลือกดีไซเนเตทพอร์ต

ดีไซเนเตทพอร์ตมีคอนเซ็ปต์ว่า ลิงค์เพียงหนึ่งเดียวในเซกเมนต์ที่ใช้ในการส่งและรับทราฟ ฟิก โดยสำหรับตัวรูทบริดจ์นั้นถือว่า ทุก ๆ พอร์ตของรูทบริดจ์จะเป็นดีไซเนเตทพอร์ตและสำหรับนอนรูทบริดจ์จะให้พอร์ตที่ต่อกับรูทพอร์ตของสวิตช์ตัวข้างเคียงเป็นดีไซเนเตทพอร์ต และสำหรับพอร์ตที่ไม่ใช่รูทพอร์ตและดีไซเนเตทพอร์ตจะถูกบล็อก (Block)

ด้วยหลักการทำงานนี้ จะทำให้แต่ละเครือข่ายแลนสามารถส่งเฟรมข้อมูลไปยังสวิตช์ได้เพียงเครื่องเดียว และเกิดสภาพของต้นไม้ (Tree) ขึ้น โดยพอร์ตที่ไม่ได้ใช้งานจะเป็นพอร์ตสำรอง และเมื่อดำเนินการทำจนเกิดเป็นสแปนนิ่งทรีแล้ว จะได้เครือข่ายที่มีลักษณะ

- **One Root Bridge per Network**

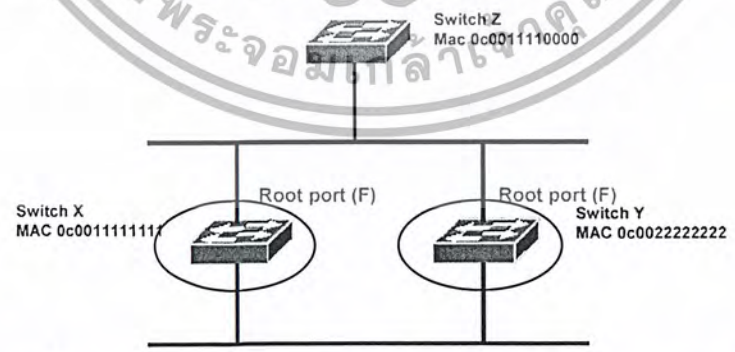
  1 Root Bridge ต่อ 1 ระบบเครือข่าย



รูปที่ 6-5 แสดงระบบเครือข่ายที่มีสวิตช์ Z เป็นรูทบริดจ์

- **One Root port per Non-Root Bridge**

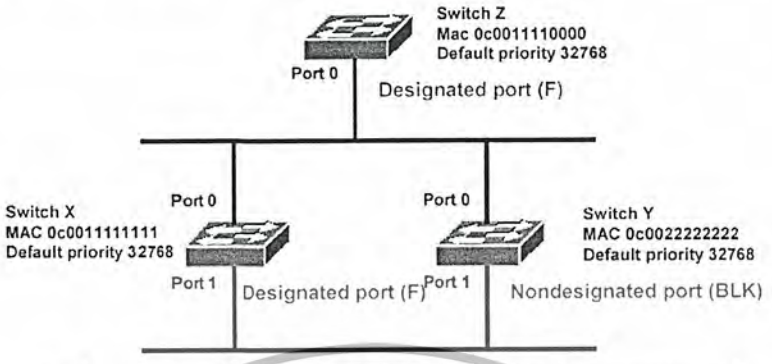
  1 Root Port ต่อ 1 Non-root Bridge



รูปที่ 6-6 แสดงระบบเครือข่ายที่มีสวิตช์ X และสวิตช์ Y เป็นนอนรูทบริดจ์

- **One Designated port per Segment**

1 Designated port ต่อ 1 เซกเมนต์



รูปที่ 6-7 แสดงระบบเครือข่ายที่มี 1 ดีไซเนเตทพอร์ตต่อ 1 เซกเมนต์

6.2 สแปนนิ่งทรีสเตท (Spanning Tree Port States)

ในการใช้สแปนนิ่งโพรโตคอลทุก ๆ พอร์ตของสวิตช์ จะต้องทำงานเป็นขั้นตอนผ่านไปในแต่ละสเตทโดยจะเริ่มต้นที่ Disabled State และสิ้นสุดที่สเตทสุดท้ายซึ่งอนุญาตให้พอร์ตสามารถรับ-ส่งเฟรมข้อมูลได้ โดยสเตทมีดังต่อไปนี้

6.2.1 Disabled – เป็นสเตทของพอร์ตที่ชัทดาวน์ (Shut down) คือไม่สามารถทำอะไรได้เลย สเตทนี้ไม่ถือว่าเป็นส่วนหนึ่งของสแปนนิ่งทรีสเตท

6.2.2 Blocking – หลังจากการติดตั้งค่าให้กับพอร์ตแล้ว พอร์ตจะเข้าสู่ Blocking State เพื่อไม่ให้การส่งเฟรมข้อมูลแบบลูป โดยในสเตทนี้จะไม่สามารถรับ – ส่งข้อมูลได้ รวมถึงไม่มีการเรียนรู้แมคแอดเดรส คือ ไม่มีการเพิ่มแมคแอดเดรสลงตารางแมคแอดเดรสอีกด้วย แต่สามารถรับ BPDU จากสวิตช์ข้างเคียงได้ นอกจานี้พอร์ตที่อยู่ในโหมดสแตนด์บาย (Standby Mode) จะเข้าสู่ Blocking State เช่นเดียวกัน

6.2.3 Listening – พอร์ตจะเปลี่ยนสเตทจาก Blocking State ไปเป็น Listening State ก็ต่อเมื่อพอร์ตได้รับการเลือกให้เป็นรูทพอร์ตหรือดีไซเนเตทพอร์ต หรือในอีกนัยหนึ่งก็คือ พอร์ตกำลังจะสามารถฟอร์เวิร์ดทราฟฟิก (Forward Traffic) ได้นั่นเอง โดยในสเตทนี้พอร์ตจะสามารถรับ – ส่ง BPDU ได้แต่ถ้าพอร์ตสูญเสียสถานะการเป็นรูทพอร์ตหรือดีไซเนเตทพอร์ต เมื่อใด พอร์ตจะกลับไปสู่ Blocking State อีกครั้ง

6.2.4 Learning – เมื่อผ่านช่วงเวลาที่เรียกว่า Forward Delay ใน Listening State แล้วพอร์ตจะเปลี่ยนไปสู่ Learning State โดยที่พอร์ตจะสามารถรับ – ส่ง BPDU ได้และสวิตช์ยังสามารถเรียนรู้แมคแอดเดรสได้อีกด้วย

6.2.5 Forwarding – เมื่อผ่านช่วงเวลาที่เรียกว่า Forward Delay ใน Learning State แล้วพอร์ตจะเปลี่ยนไปสู่ Forwarding State โดยพอร์ตจะสามารถรับ-ส่งข้อมูล, เรียนรู้แมคแอดเดรสและรับ-ส่ง BPDU ได้โดยในสเตทนี้ถือว่าได้ทำสแปนนิ่งทรีเสร็จสิ้นแล้วนั่นเอง

6.3 ชนิดของสแปนนิ่งโพรโตคอล (Types of Spanning Tree Protocol)

ในตอนเริ่มต้นนั้นสแปนนิ่งโพรโตคอลได้รับการพัฒนามาเพื่อใช้กับเครือข่ายแลน (VLAN) เดียวเท่านั้น การสร้างสแปนนิ่งโพรโตคอลเพื่อให้สามารถใช้ได้กับ Multiple VLANs นั้นจำเป็นต้องประกอบไปด้วยหลาย ๆ สิ่ง ด้วยเหตุนี้ จึงได้มีการสร้างสแปนนิ่งโพรโตคอลที่แตกต่างกันออกมา ซึ่ง ณ ที่นี้เราจะขอหยิบขึ้นมา 3 ประเภทด้วยกันคือ

6.3.1 คอมมอนสแปนนิ่งทรี (Common Spanning Tree (CST))

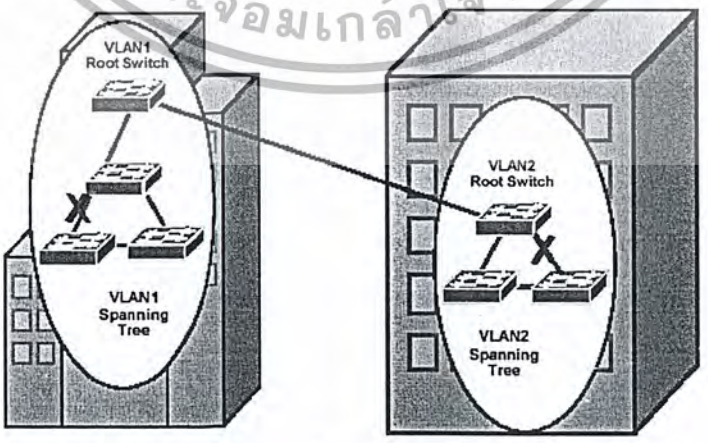
มาตรฐาน IEEE 802.1Q ได้กำหนดถึงการสร้างทรังค์ลิงค์ (Trunk Link) ที่ใช้ระหว่างสวิตช์ และได้กำหนดให้ใช้เป็น 1 สแปนนิ่งทรีต่อหนึ่งเครือข่าย ก็คือทุกวีแลนมีชื่อเรียกว่าคอมมอนสแปนนิ่งทรี (Common Spanning Tree (CST)) หรือ โมโนสแปนนิ่งทรี (Mono Spanning Tree (MST)) ทุก ๆ BPDUs จะถูกส่งไปทั่วทั้งเครือข่าย

การใช้เพียง 1 สแปนนิ่งทรีต่อหลาย ๆ วีแลน มีข้อดีคือ สามารถใช้คำสั่งได้ง่าย ไม่ยุ่งยากและยังลดการใช้ซีพียู (CPU) ของสวิตช์ในการคำนวณอีกด้วย อย่างไรก็ตาม การกระทำเช่นนี้ยังมีข้อเสียอยู่ก็คือ Redundant Link ที่ได้กำหนดขึ้นจะไม่ได้ใช้งาน ถูกบล็อกเพื่อกำจัดลูปและยังมีข้อจำกัดในเรื่องวีแลนคือพอร์ตที่ถูกบล็อค อาจจะเป็นพอร์ตที่ใช้ส่งเฟรมข้อมูลของวีแลนใด ๆ ก็ได้ จึงอาจจะเป็นเหตุให้ไม่สามารถส่งเฟรมข้อมูลได้

6.3.2 เปอร์วีแลนสแปนนิ่งทรี (Per-VLAN Spanning Tree (PVST))

เปอร์วีแลนสแปนนิ่งทรี (Per-VLAN Spanning Tree) เป็นสแปนนิ่งทรีที่มีความยืดหยุ่นมากกว่า CST เนื่องจากมีหลักการทำงานในการสร้างสแปนนิ่งทรีแยกเป็นของแต่ละวีแลนเลย เพื่อสามารถใช้คำสั่งได้อย่างเป็นอิสระ และยังมีประสิทธิภาพมากกว่า Multiple Spanning Tree สามารถสร้าง Load Balancing บน Redundant Links เมื่อลิงค์นั้นถูกกำหนดไว้ในคนละวีแลนกัน

เนื่องจากคุณสมบัติของ PVST จึงจำเป็นต้องใช้ Cisco Inter-Switch Link (ISL) ในการส่งข้อมูลผ่านทรังค์ลิงค์ระหว่างสวิตช์

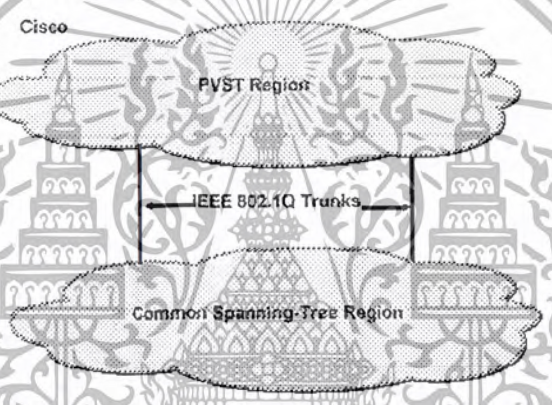


รูปที่ 6-8 แสดงตัวอย่างเครือข่ายที่ใช้เปอร์วีแลนสแปนนิ่งทรี

6.3.3 เปอร์วีแลนสแปนนิ่งทรีพลัส (Per-VLAN Spanning Tree Plus (PVST+))

เปอร์วีแลนสแปนนิ่งทรีพลัส (Per-VLAN Spanning Tree Plus) มีความสามารถในการติดต่อสื่อสารระหว่าง CST และ PVST PVST+ สามารถสนับสนุนการทำงานของสแปนนิ่งทรีทั้ง 3 กลุ่มคือ Catalyst ที่ใช้ PVST, Catalyst ที่ใช้ PVST+ และสวิตช์ที่ใช้งาน CST/MST บน 802.1Q

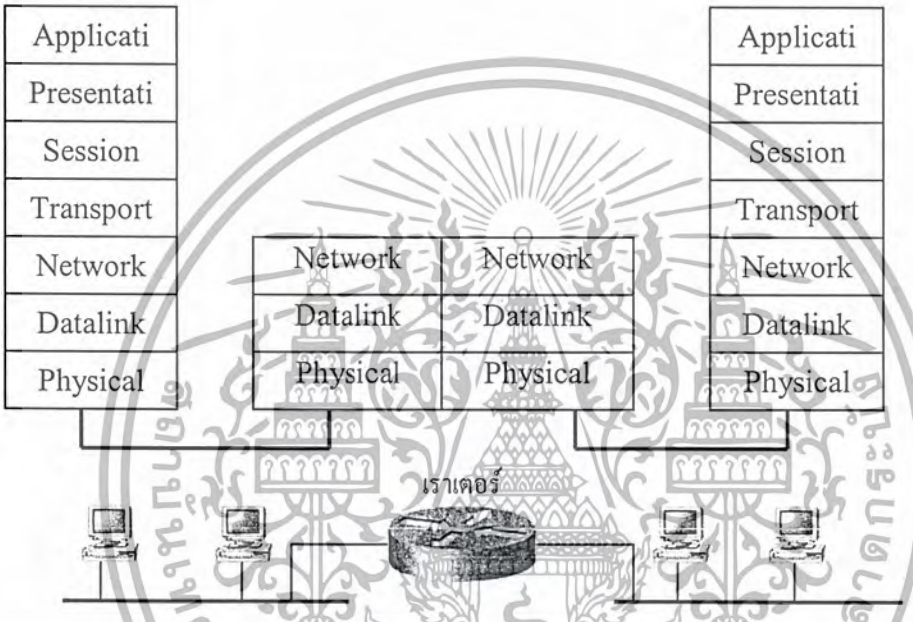
หลักในการทำงานคือ PVST+ จะเป็นเสมือนตัวที่ใช้ติดต่อสื่อสารระหว่างกลุ่มของ CST สวิตช์และกลุ่มของ PVST สวิตช์โดย PVST+ สามารถติดต่อกับ PVST ได้โดยตรงโดยผ่าน ISL Trunks และติดต่อกับ CST อย่างไรก็ตาม PVST+ จะแลกเปลี่ยน BPDUs กับ CST บนวีแลน1 สำหรับ BPDUs ที่มาจากสแปนนิ่งทรีของแต่ละวีแลนนั้นจะเดินทางไปยังส่วนของ CST ในระบบเครือข่ายด้วย Tunnel PVST+ จะส่ง BPDUs ที่กล่าวมานี้โดยการใช้ Multicast Address ดังนั้น CST สวิตช์จะสามารถ Forward BPDUs ไปยังสวิตช์ข้างเคียงได้ จนในที่สุด BPDUs จึงจะสามารถเดินทางไปถึง PVST+ สวิตช์ได้



รูปที่ 6-9 แสดงการทำงานของเปอร์วีแลนสแปนนิ่งทรีพลัส

บทที่ 7
เราเตอร์

เราเตอร์เป็นอุปกรณ์ที่ทำงานวนระดับชั้นเน็ตเวิร์กตามรูปที่ 7-1 เราเตอร์ทำงานร่วมกับฮาร์ดแวร์ในระดับดาต้าลิงค์ได้หลายรูปแบบ หน้าที่ของเราเตอร์คือจัดแบ่งเครือข่ายและเลือกเส้นทางที่เหมาะสมเพื่อนำส่งแพ็กเก็ต เราเตอร์จะป้องกันการบรอดคาสต์แพ็กเก็ตจากเครือข่ายหนึ่งไม่ให้ข้ามมาอีกเครือข่ายหนึ่ง

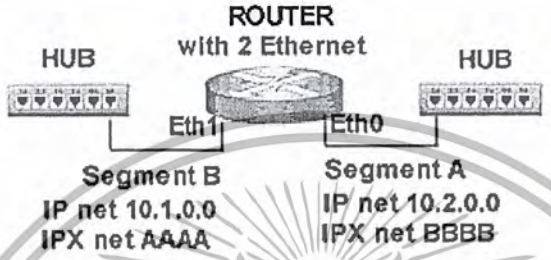


รูปที่ 7-1 แบบจำลองการทำงานของเราเตอร์

ในอินเทอร์เน็ตมักเรียกเราเตอร์ว่า ไอพีเราเตอร์ (IP Router) เนื่องจากเราเตอร์ทำงานตามข้อกำหนดของโพรโตคอลไอพี เราเตอร์ทำหน้าที่เลือกเส้นทางโดยสร้างแผนที่เครือข่ายและเก็บอยู่ในรูปตารางเส้นทาง เมื่อเราเตอร์ได้รับแพ็กเก็ตก็จะตรวจสอบแอดเดรสปลายทางและส่งแพ็กเก็ตไปยังอีกอินเทอร์เฟสที่เป็นช่องทางไปสู่เครือข่ายปลายทาง เราเตอร์ประกอบด้วยหลายอินเทอร์เฟสเพื่อเชื่อมต่อกับเครือข่ายต่างชนิดเข้าด้วยกันได้ ตัวอย่างเช่น อินเทอร์เฟสอีเทอร์เน็ต โทเค็นริง เอฟดีดีไอ เอทีเอ็ม หรืออินเทอร์เฟสแบบอนุกรม เป็นต้น

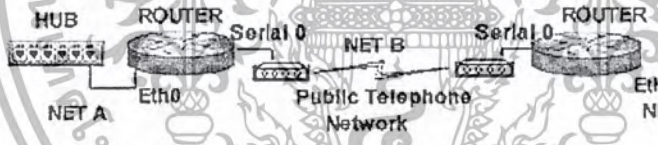
เราเตอร์เป็นอุปกรณ์ที่ช่วยให้แพ็กเก็ตของโปรโตคอลต่าง ๆ สามารถวิ่งผ่านไปถึงจุดหมายปลายทางตามที่ต้องการได้ โดยอาศัยค่าจากตาราง Routing(Routing Table) เป็นตัวช่วยในการพิจารณาเลือกเส้นทาง เพื่อให้แพ็กเก็ตวิ่งไปหาจุดหมายปลายทางในเส้นทางที่เหมาะสม ในตัวเราเตอร์จะมีการใช้โปรโตคอลสองประเภทคือแบบ Static และแบบ Dynamic (เช่น RIP, OSPF, IGRP, EGP, BGP) กรณีที่ใช้ Routing โปรโตคอลแบบ Dynamic เราเตอร์แต่ละตัวจะมีการแลกเปลี่ยนค่าในตาราง Routing ซึ่งกันและกันตามช่วงเวลาที่กำหนด ลักษณะการนำเราเตอร์ไปใช้งานจะมีอยู่ 2 ลักษณะคือ

แบบที่หนึ่งใช้สำหรับเชื่อมระหว่างแลน 2 เซกเมนต์ดังรูปที่ 1 จุดประสงค์ของการใช้งานในลักษณะนี้มีหลายอย่าง เช่น ลด Traffic ในเครือข่ายขนาดใหญ่ให้ลดลงด้วยการแบ่งเครือข่ายออกเป็น 2 เซกเมนต์หรือมากกว่า โดยใช้เราเตอร์เป็นตัวคั่นระหว่างเซกเมนต์ ซึ่งจะช่วยให้การส่งแพ็กเก็ตแบบ Broadcast ถูกจำกัดอยู่ภายในเซกเมนต์เท่านั้น และช่วยกั้นไม่ให้แพ็กเก็ตที่ต้องการคุยกันภายในเซกเมนต์ไม่ให้ข้ามไปรบกวนเซกเมนต์อื่น หรือกรณีเมื่อขอ IP Address จาก ISP (Internet Service Provider) เพื่อใช้ติดต่อกับเครือข่ายอินเทอร์เน็ตอยู่หนึ่งคลาสแต่ต้องการแบ่งให้หน่วยงานต่าง ๆ เป็นเครือข่ายย่อย (Sub Network) ท่านต้องใช้เราเตอร์เป็นตัวคั่นระหว่างเครือข่ายย่อย



รูปที่ 7-2 เราเตอร์เชื่อมแลน 2 เซกเมนต์

แบบที่สองใช้สำหรับเชื่อม 2 เครือข่ายที่อยู่ห่างกันเกินความสามารถของมาตรฐานในสาย 10Base5 (500 เมตร), Wireless Lan (ใช้คลื่นวิทยุ) หรือสายเส้นใยนำแสง โดยจะใช้สายเคเบิลโทรศัพท์ในการเชื่อม 2 เครือข่ายดังรูปที่ 2



รูปที่ 7-3 เราเตอร์เชื่อมแลน 2 เน็ตเวิร์กที่อยู่ห่างไกลกัน

เราเตอร์ประกอบด้วยส่วนสำคัญ 2 ส่วน คือ ฮาร์ดแวร์และซอฟต์แวร์ระบบปฏิบัติการ ด้านหลังของเราเตอร์ประกอบด้วยพอร์ต Ethernet ซึ่งนิยมใช้เป็น RJ45 สำหรับต่อสาย UTP ไปเชื่อมต่อกับฮับหรือสวิตชิง ในเราเตอร์รุ่นใหม่ ๆ จะใช้ตัวเชื่อมต่อเป็นแบบ FastEthernet ซึ่งสามารถเลือกความเร็วได้ว่าจะใช้ความเร็วของ Ethernet เป็น 10 MB หรือ 100 MB หรือ Gigabit Ethernet ที่สามารถส่งข้อมูลได้ถึง 1000 MB เพื่อให้เหมาะสมกับเครือข่ายของท่าน และซอฟต์แวร์ระบบปฏิบัติการที่ให้มาในเราเตอร์อาจไม่มีความสามารถบางอย่างที่ต้องการ ท่านสามารถอัพโหลดซอฟต์แวร์ระบบปฏิบัติการตัวใหม่ที่มีความสามารถตามที่ต้องการเข้าไปได้ แต่ควรจะตรวจสอบรุ่นและหน่วยความจำ (RAM) ของเราเตอร์ว่าสามารถใช้กับซอฟต์แวร์ตัวใหม่ได้หรือไม่

บทที่ 8
โพรโตคอลเลือกเส้นทาง (Routing Protocol)

การเลือกเส้นทางเป็นหน้าที่สำคัญอย่างหนึ่งของไอพี เราเตอร์อาศัยตารางเส้นทาง (Routing Table) เพื่อนำส่งดาต้าแกรมไปยังเครือข่ายปลายทาง ตารางเส้นทางอาจปรับปลี่ยนได้ตามสภาพเครือข่ายอัติโนมัติโดยใช้โพรโตคอลเลือกเส้นทาง หรือผู้ดูแลระบบเป็นคนปรับเปลี่ยนเส้นทางเอง

8.1 การเลือกเส้นทาง (Routing)

การเลือกเส้นทางเป็นกระบวนการที่เกิดขึ้นในระดับชั้นที่ 3 หรือระดับเน็ตเวิร์ก ของแบบจำลองทีซีพี/ไอพี ซอฟต์แวร์ไอพีที่อยู่ในโฮสต์หรือเราเตอร์จะนำส่งดาต้าแกรมไปตามเส้นทาง โดยอาศัยเลขเครือข่ายของไอพีแอดเดรสตามแต่ละคลาส เลขเครือข่ายเป็นเสมือนค่ากำหนดตำแหน่งปลายทางของเครือข่ายซึ่งคล้ายกับรหัส 3 ตัวแรกที่กำหนดชุมสายโทรศัพท์ ระบบรหัสชุมสายจะกำหนดที่ตั้งทางภูมิศาสตร์ด้วยว่าอยู่ทิศทางใด แต่เลขเครือข่ายในไอพีแอดเดรสนี้ไม่ได้มีส่วนสัมพันธ์กับที่ตั้งของเครือข่าย

เครือข่ายโดยทั่วไปประกอบด้วยสถานีส่วนปลาย (End Node) และเราเตอร์หรืออุปกรณ์อื่นทำงานร่วมกัน โดยกระบวนการเลือกหาเส้นทางจะเกิดขึ้นทั้งที่สถานีส่วนปลายและที่เราเตอร์ คือ

1. สถานีส่วนปลายที่เป็นสถานีส่งต้องทราบว่าจะนำส่งดาต้าแกรมให้เราเตอร์ได้อย่างไรและเมื่อใด
2. เราเตอร์ต้องทราบเส้นทางเชื่อมโยงไปยังเราเตอร์ตัวอื่น เพื่อส่งดาต้าแกรมไปตามเส้นทางที่เหมาะสมที่สุด
3. เราเตอร์ที่เชื่อมกับเครือข่ายของสถานีปลายทางต้องทราบถึงวิธีส่งดาต้าแกรมไปยังสถานีปลายทางนั้น

ในตอนแรกสถานีต้นทางจะเป็นผู้ตัดสินใจขั้นแรกว่าต้องส่งดาต้าแกรมไปยังสถานีในเครือข่ายด้วยกันเองหรือต้องส่งผ่านเราเตอร์ โดยสถานีต้นทางจะเปรียบเทียบเลขเครือข่ายของแอดเดรสต้นทางและปลายทางกับค่าซับเน็ตมาสก์ หากได้เลขครือข่ายเหมือนกันแสดงว่าสถานีปลายทางอยู่ในเครือข่ายเดียวกัน สถานีต้นทางจะใช้เออาร์พีสอบถามฮาร์ดแวร์แอดเดรสหรืออ่านจากแคช และบรรจุฮาร์ดแวร์แอดเดรสเข้าสู่เฟรมดาต้าลิงค์เพื่อส่งตรงไปยังสถานีปลายทาง แต่ถ้าเลขเครือข่ายมีค่าต่างกันแสดงว่าสถานีปลายทางอยู่ต่างเครือข่ายกัน สถานีส่งก็จะส่งเฟรมไปให้เราเตอร์เพื่อให้เราเตอร์นำส่งต่อไป เมื่อเราเตอร์ได้รับเฟรมนี้ก็จะส่งต่อไป ไอพีแอดเดรสต้นทางและปลายทางในไอพีดาต้าแกรมจะไม่เปลี่ยนแปลงค่าระหว่างการลำเลียง แต่ฮาร์ดแวร์แอดเดรสจะเปลี่ยนแปลงไปตามเครือข่าย

8.2 ตารางเส้นทาง(Routing Table)

โฮสต์และเราเตอร์จะเก็บแอดเดรสปลายทางสำหรับใช้เป็นเส้นทางส่งดาต้าแกรมไว้ในรูปตารางที่เรียกว่า ตารางเส้นทาง (Routing Table) ค่าในตารางเส้นทางมักประกอบด้วยไอพีแอดเดรสของเครือข่ายปลายทาง และเกตเวย์ซึ่งเป็นทางออกของดาต้าแกรม

ตารางเส้นทางจะบรรจุแอดเดรสหนึ่งซึ่งทำหน้าที่เป็นช่องทางออกไปสู่เครือข่ายใดๆ ที่ไม่ได้ระบุอยู่ในตารางเส้นทาง แอดเดรสนี้เรียกว่า ดีฟอลต์เกตเวย์ (Default Gateway) หรือ ดีฟอลต์เราเตอร์ (Default Router) ซึ่งแทนด้วยแอดเดรส 0.0.0.0 ไอพีจะเลือกเส้นทางโดยตรวจสอบว่าแอดเดรสปลายทางตรงกับรายการใดในตาราง แล้วส่งดาต้าแกรมไปยังเกตเวย์ของรายการนั้น หากไม่พบรายการใดในตาราง ก็จะส่งไปยังดีฟอลต์เกตเวย์

8.3 ประเภทของการเลือกเส้นทาง

เมื่อดาต้าแกรมเดินทางออกนอกเครือข่ายที่อาจต้องผ่านเราเตอร์จำนวนมากโดยมีเส้นทางลำเลียงได้หลายเส้นทาง ปัญหาที่สำคัญคือเราเตอร์จะทราบได้อย่างไรว่ามีเส้นทางใดบ้าง และเส้นทางใดที่เป็นเส้นทางที่ดีที่สุด

เราเตอร์จะส่งดาต้าแกรมได้จำเป็นต้องมีแผนที่เครือข่าย ซึ่งก็คือตารางเส้นทาง สำหรับวิธีที่ใช้ในการสร้างตารางเส้นทางนี้มี 2 แบบคือ

1. การเลือกเส้นทางแบบสแตติก(Static Routing) ตารางเลือกเส้นทางสร้างขึ้นและแก้ไขโดยผู้ดูแลระบบเครือข่าย
2. การเลือกเส้นทางแบบไดนามิก(Dynamic Routing) ใช้ซอฟต์แวร์คำนวณหาค่าตารางเลือกเส้นทาง ตารางสามารถปรับเปลี่ยนได้หากสภาพเครือข่ายเปลี่ยนไป

8.3.1 การเลือกเส้นทางแบบสแตติก

การเลือกเส้นทางแบบนี้ผู้ดูแลระบบเครือข่ายเป็นผู้พิจารณาและคำนวณหาเส้นทางทั้งหมด โดยใส่ค่าตารางเส้นทางให้กับเราเตอร์ทุกตัว ตารางเลือกเส้นทางนี้จะมีค่าตายตัวตลอดถึงแม้ว่าสภาพของเครือข่ายจะเปลี่ยนไปดังนั้นผู้ดูแลระบบจะต้องคอยตรวจสอบเครือข่ายและปรับเปลี่ยนตารางเส้นทางให้ถูก

รูปที่ 5-1 แสดงเครือข่ายที่ประกอบด้วยเราเตอร์ 2 ตัว เราเตอร์แต่ละตัวจะมีตารางเส้นทางที่ผู้ดูแลระบบป้อนเพื่อใช้กำหนดทิศทางการส่งดาต้าแกรม ทุกเครือข่ายที่เชื่อมโดยตรงกับเราเตอร์อื่นจะมีค่าเกตเวย์เท่ากับไอพีแอดเดรสประจำอินเทอร์เฟสนั้น ส่วนเครือข่ายที่ต้องผ่านเราเตอร์อื่นจะมีค่าเกตเวย์เท่ากับไอพีแอดเดรสของเราเตอร์ขั้นถัดไป (Next Hop Router) เช่นตารางเส้นทางในเราเตอร์ R1 มีค่าเกตเวย์ประจำเครือข่าย 1 และเครือข่าย 2 เท่ากับ 1.1 และ 2.1 ตามลำดับ ส่วนเกตเวย์สำหรับเครือข่าย 4 จะมีค่าเท่ากับ 3.2 ซึ่งเป็นแอดเดรสประจำอินเทอร์เฟส e1 ของ R2 (R2 เป็นเราเตอร์ขั้นถัดไปของ R1)

การเลือกเส้นทางแบบสแตติกนิยมใช้กับการเชื่อมโยงแบบจุดต่อจุดระหว่างเราเตอร์สองตัว เช่น เครือข่ายมีทางออกไปสู่ภายนอกหรืออินเทอร์เน็ตเพียงช่องทางเดียว มักจะกำหนดเส้นทางแบบสแตติก การเลือกเส้นทางแบบสแตติกมีข้อดีข้อเสียดังนี้

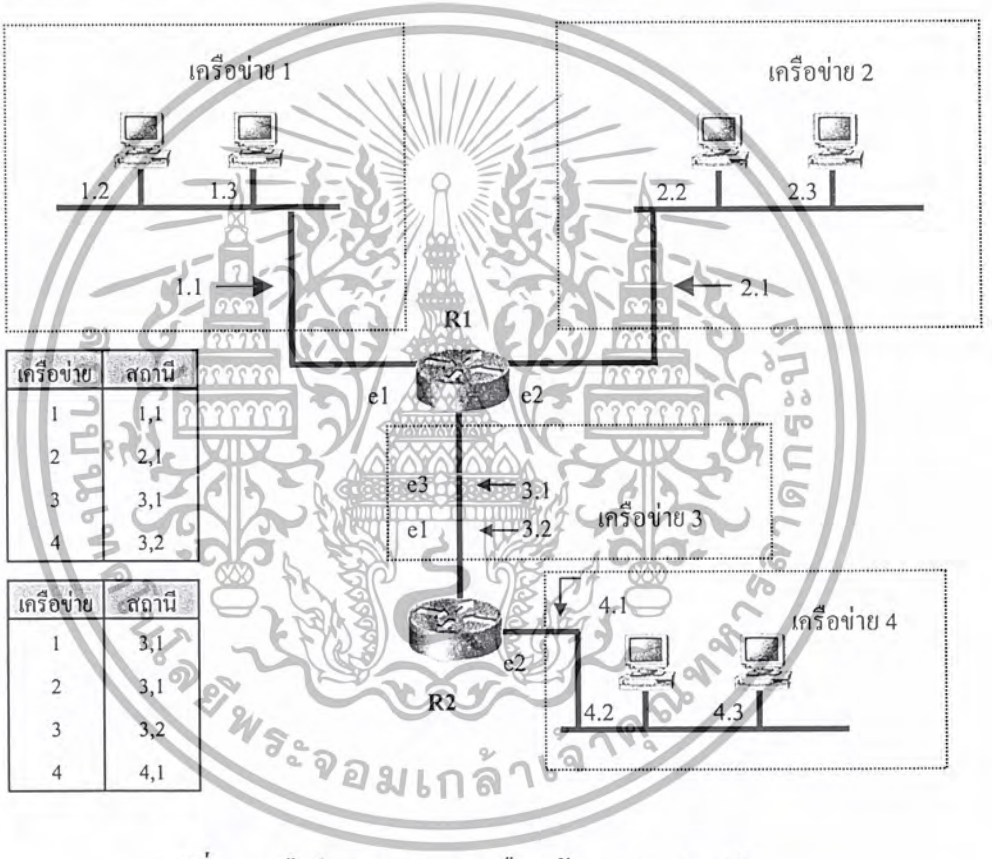
<u>ข้อดี</u>

1. สะดวกต่อการใช้งานกับเครือข่ายขนาดเล็ก
2. ไม่ต้องใช้ซอฟต์แวร์เลือกเส้นทาง เราเตอร์ไม่จำเป็นต้องมีซีพียูสมรรถนะสูง
3. ประหยัดแบนด์วิดเครือข่ายเนื่องจากไม่ต้องแลกเปลี่ยนข้อมูลระหว่างเราเตอร์

4. สามารถจำกัดให้เข้าถึงได้เฉพาะในเครือข่ายที่ต้องการ

ข้อเสีย

1. ไม่เหมาะกับเครือข่ายขนาดใหญ่ เพราะผู้ดูแลระบบสามารถคำนวณหาเส้นทางในเครือข่ายที่มีขนาดเล็กได้ แต่ในเครือข่ายขนาดใหญ่ที่มีเราเตอร์เป็นจำนวนมากการคำนวณและป้อนค่าเข้าสู่เราเตอร์โดยตรงเป็นสิ่งที่เกินวิสัย
2. ไม่สะดวกต่อการเปลี่ยนโครงสร้างของเครือข่าย เพราะผู้ดูแลต้องคำนวณหาเส้นทางใหม่ทุกครั้งที่มีการเปลี่ยนโครงสร้างของเครือข่าย
3. ตารางเส้นทางเป็นตารางคงตัวไม่สามารถเปลี่ยนแปลงได้เอง ถ้าเส้นทางใดถูกตัดขาดไป ผู้ดูแลระบบจะต้องคอยตรวจสอบและแก้ไขเอง



รูปที่ 8-1 เครือข่ายการแสดงการเลือกเส้นทางแบบสแตติก

8.3.2 การเลือกเส้นทางแบบไดนามิก

การเลือกเส้นทางแบบนี้จะใช้ซอฟต์แวร์ทำหน้าที่แลกเปลี่ยนข้อมูลการเลือกเส้นทางระหว่างเราเตอร์ด้วยกัน โดยใช้โพรโตคอลเลือกเส้นทาง เราเตอร์จะสร้างตารางเลือกเส้นทางจากสภาพเครือข่ายขณะนั้น หากเครือข่ายมีการเปลี่ยนแปลงตารางเส้นทางก็จะเปลี่ยนแปลงตามไปด้วย

การเลือกเส้นทางแบบไดนามิกต้องอาศัยการแลกเปลี่ยนค่าเส้นทางระหว่างเราเตอร์และใช้ซีพียูในเราเตอร์เพื่อสร้างตารางเส้นทาง เราเตอร์ประเภทนี้จึงมักมีราคาสูงกว่าเราเตอร์ที่มีโพรโตคอลแบบสแต

ติกอย่างเดียว เพราะต้องออกแบบซอฟต์แวร์ให้ทำงานตามโพรโตคอลเลือกเส้นทาง และซีพียูต้องมีสมรรถนะสูงพอในการคำนวณหาตารางเส้นทาง

ข้อดีของการการเลือกเส้นทางแบบไดนามิก

1. รองรับขนาดเครือข่ายที่ขยายขึ้นเป็นลำดับได้
2. ตารางเส้นทางเปลี่ยนแปลงค่าเองตามการทำงานของซอฟต์แวร์ เส้นทางใดที่ถูกตัดขาดจะมีการหาเส้นทางใหม่ทดแทน

ประเภทโพรโตคอลเลือกเส้นทางแบบไดนามิก

โพรโตคอลเลือกเส้นทางแบบไดนามิกสามารถจัดแยกประเภทออกได้หลายรูปแบบ ในที่นี้จะกล่าวเพียง 2 รูปแบบคือ

1. โพรโตคอลเกตเวย์ภายนอกและภายใน (Interior Gateway Protocol และ Exterior Gateway Protocol)
2. โพรโตคอลดิสแทนซ์เวกเตอร์และลิงค์สเกต (Distance Vector Protocol และ Link State Protocol)

8.3.2.1 โพรโตคอลเกตเวย์ภายนอกและภายใน

การจัดแบบนี้แบ่งชนิดโพรโตคอลออกเป็นโพรโตคอลเกตเวย์ภายนอกและโพรโตคอลเกตเวย์ภายใน โพรโตคอลเกตเวย์ภายนอกมีหน้าที่แลกเปลี่ยนข้อมูลเส้นทางระหว่างเครือข่ายที่มีการบริหารงานเป็นอิสระออกจากกัน โดยแต่ละเครือข่ายที่มีการบริหารเป็นอิสระออกจากกันจะเรียกว่า ระบบออโตโนมัส (Autonomous System) แต่ละระบบออโตโนมัสมีหมายเลยประจำของตนเองเรียกว่า เลขระบบออโตโนมัส (Autonomous System Number) หมายเลขนี้สามารถขอได้จากหน่วยงานนิก (NIC) ประจำภูมิภาค เลขระบบออโตโนมัสเป็นค่าที่ระบุว่าข้อมูลเส้นทางที่แลกเปลี่ยนระหว่างเครือข่ายนั้นมาจากที่ใด

อินเทอร์เน็ตในยุคแรกใช้โพรโตคอล อีจีพี (EGP: Exterior Gateway Protocol) เป็นโพรโตคอลเกตเวย์ภายนอก แต่ในปัจจุบันโพรโตคอลที่นิยมใช้ระหว่างเครือข่ายคือบีจีพี (BGP : Border Gateway Protocol) และนำมาใช้งานแทนอีจีพี พีจีพีผ่านการพัฒนามาเป็นลำดับกระทั่งปัจจุบันเป็นรุ่นที่ 4 จึงเรียกว่า บีจีพี-4

โพรโตคอลเกตเวย์ภายในเป็นโพรโตคอลที่ออกแบบเพื่อใช้งานในระบบออโตโนมัสหนึ่งๆ เช่น อาร์ไอพี (RIP : Routing Information Protocol) และ โอเอสพีเอฟ (OSPF: Open Shortest Path First) โพรโตคอลทั้งสองนี้เป็นที่ยอมรับเป็นมาตรฐานสากลและใช้งานอย่างแพร่หลายในเครือข่ายทั่วไป ในขณะที่โพรโตคอลเฉพาะที่ออกแบบโดยบริษัทซิสโก้ (Cisco) คืออีไอจีอาร์พี (EIGRP : Enhanced Interior Gateway Routing Protocol) เป็นอีกโพรโตคอลหนึ่งที่นิยมใช้ตามความแพร่หลายของเราเตอร์ของซิสโก้

เมตริก (Metric)

เมตริกเป็นค่าที่นำมาใช้คำนวณหาว่าเส้นทางใดเหมาะต่อการใช้มากกว่าเส้นทางอื่น ค่าเมตริกที่ใช้อาจเป็นได้ทั้งระยะทาง เวลาหน่วง ความน่าเชื่อถือ หรือความเร็ว โพรโตคอลที่ไม่สลับซับซ้อนอาจจะ

เลือกใช้เมตริกเพียงประเภทใดประเภทหนึ่ง เช่นอาจใช้เฉพาะระยะทางซึ่งนับจากจำนวนเราเตอร์ที่ต้องส่งผ่านหรือเรียกว่าจำนวนขั้น (Hop Count)

จำนวนขั้นนับจากจำนวนเราเตอร์ที่ดาต้าแกรมต้องเดินผ่านดังตัวอย่างในรูปที่ 5-2 หากดาต้าแกรมไม่ต้องเดินทางผ่านเราเตอร์จำนวนขั้นจะเท่ากับ 0 หากต้องข้ามเราเตอร์หนึ่งตัว จำนวนขั้นจะเพิ่มขึ้นทีละ 1 แต่โพรโตคอลบางชนิดอาจนับจำนวนขั้นตามจำนวนลิงค์ที่ดาต้าแกรมเดินทางผ่านแทนการนับด้วยจำนวนเราเตอร์ หากเครือข่ายอยู่ติดกับเราเตอร์จะมีจำนวนขั้นเท่ากับ 1 แทนที่จะเท่ากับ 0

8.3.2.2 โพรโตคอลดิสแทนซ์เวกเตอร์และลิงค์สเทต

การจัดแบ่งโพรโตคอลในแบบนี้อาศัยรูปแบบของข้อมูลที่ส่งผ่านระหว่างเราเตอร์และวิธีที่เราเตอร์สร้างตารางเลือกเส้นทางจากข้อมูลนั้น

เราเตอร์ที่ใช้โพรโตคอลดิสแทนซ์เวกเตอร์อาศัยระยะทางเพื่อกำหนดว่าเส้นทางใดเหมาะสมกว่าเส้นทางอื่น ความหมายของดิสแทนซ์เวกเตอร์คือใช้ระยะทางเป็นค่าเมตริก และแอดเดรสของเครือข่ายปลายทางเป็นเวกเตอร์กำหนดจุดหมายปลายทาง

การทำงานพื้นฐานของโพรโตคอลดิสแทนซ์เวกเตอร์จะส่งข้อมูลเลือกเส้นทางไปยังเราเตอร์ที่อยู่ข้างเคียงทุกตัวอย่างสม่ำเสมอเป็นช่วงเวลา ข้อมูลเลือกเส้นทางประกอบด้วยตารางเลือกเส้นทางของตนเองทั้งหมดที่กำกับด้วยเมตริก เราเตอร์แต่ละตัวจะใช้ตารางเส้นทางของตัวเองร่วมกับตารางเส้นทางที่ได้รับมาใหม่เพื่อคำนวณหาระยะทางที่สั้นที่สุด โพรโตคอลดิสแทนซ์เวกเตอร์ที่นิยมใช้ในปัจจุบันได้แก่ อาร์ไอพี ค่าเมตริกที่ใช้คือจำนวนขั้น อาร์ไอพีถือว่าเส้นทางที่ดีที่สุดคือเส้นทางที่มีจำนวนขั้นน้อยที่สุด

โพรโตคอลลิงค์สเทตไม่ได้แลกเปลี่ยนตารางเส้นทางโดยตรงเหมือนกับที่ใช้ในดิสแทนซ์เวกเตอร์ หากแต่เราเตอร์แต่ละตัวจะตรวจสอบ สถานะลิงค์ (Link State) ที่เชื่อมไปยังเราเตอร์ข้างเคียงว่าใช้งานได้หรือไม่พร้อมกับค่าเมตริกซึ่งโดยทั่วไปแล้วเป็นความเร็วของสายสื่อสาร เช่นให้ค่า 1 สำหรับสายสื่อสารที่มีความเร็ว 2 เมกะบิตต่อวินาที และ 10 สำหรับสายสื่อสารที่มีความเร็ว 9600 บิตต่อวินาที โพรโตคอลลิงค์สเทตที่นิยมใช้คือโอเอสพีเอฟ

8.4 อาร์ไอพี(RIP)

อาร์ไอพีมีกำเนิดมาจากโพรโตคอลเลือกเส้นทางในระบบเครือข่ายซีร็อกซ์ซึ่งต่อมาได้พัฒนาเป็นอาร์ไอพีและโพรโตคอลอื่นได้แก่ไอพีเอกซ์ (โนเวลล์) และไอจีอาร์พี (ซิสโก้) อาร์ไอพีเป็นโพรโตคอลแบบดิสแทนซ์เวกเตอร์และใช้ระยะทางเป็นค่าเมตริกเพื่อหาเส้นทางที่ดีที่สุดสำหรับการเลือกเส้นทาง

อาร์ไอพีเป็นโพรโตคอลแบบดิสแทนซ์เวกเตอร์ ขั้นตอนพื้นฐานที่อาร์ไอพีใช้คือขั้นตอนวิธีของฟอร์ด/ฟูเกอร์สัน หรือเรียกอีกชื่อว่าเบลแมน-ฟอร์ด เราเตอร์อาร์ไพพีจะเก็บตารางเส้นทางซึ่งประกอบด้วยแอดเดรสเครือข่าย แอดเดรสของเราเตอร์ถัดไปซึ่งเป็นเกตเวย์ไปยังเครือข่ายนั้น และเมตริกประจำเส้นทางซึ่งโดยปกติจะนับตามจำนวนเราเตอร์ระหว่างทาง

8.4.1 การทำงานของอาร์ไอพี

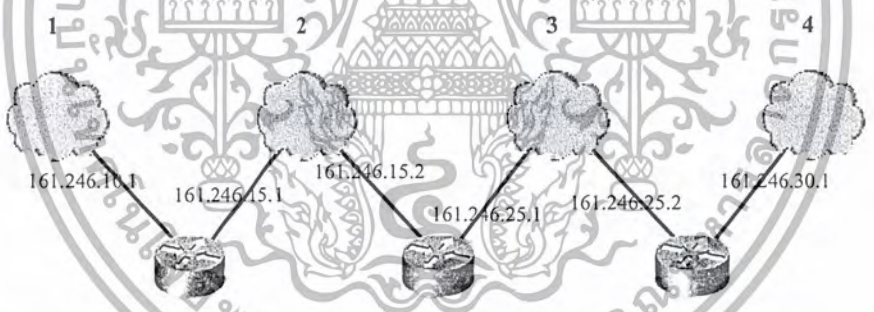
เริ่มแรกเราเตอร์แต่ละตัวจะได้รับการกำหนดแอดเดรสเครือข่ายประจำแต่ละอินเทอร์เฟสของเราเตอร์นั้น ซึ่งเราเตอร์ก็จะสร้างตารางเส้นทางจากแอดเดรสเครือข่ายที่มี แล้วส่งตารางเส้นทางของตัวเองไปให้เราเตอร์ข้างเคียงด้วยการบรอดคาสต์

เมื่อเราเตอร์ได้รับตารางเส้นทางจากเราเตอร์ตัวอื่น ก็จะปรับตารางเส้นทางของตัวเองพร้อมทั้งบรอดคาสต์ตารางของตัวเองไปให้เราเตอร์ข้างเคียงอีกเช่นกัน วิธีนี้ทำให้เราเตอร์แต่ละตัวสามารถคำนวณระยะทางและทิศทางของเครือข่ายอื่นที่เราเตอร์นั้นไม่ได้เชื่อมต่อได้ ในที่สุดเราเตอร์ทุกตัวก็จะทราบแอดเดรสทั้งหมดของเครือข่าย สำหรับการปรับค่าตารางเส้นทางเราเตอร์จะพิจารณาโดย

1. ถ้าเป็นเส้นทางใหม่ที่ไม่มีในตารางเส้นทาง จะใส่เส้นทางนั้นเข้าตารางเลย
2. ถ้าเป็นเส้นทางที่มีข้อมูลอยู่ในตารางเส้นทางแล้ว เราเตอร์จะพิจารณาว่าเป็นเส้นทางที่สั้นกว่าข้อมูลที่มีอยู่ในตาราง จะแทนที่ข้อมูลในตารางด้วยข้อมูลใหม่
3. ถ้าได้รับข้อมูลเส้นทางจากเราเตอร์ R ใดๆ และตรวจพบว่าในตารางมีเส้นทางซึ่งเราเตอร์ R เป็นเกตเวย์อยู่แล้ว ให้ปรับค่าเส้นทางใหม่ตามค่าที่ได้รับจากเราเตอร์ R นั้น

ตัวอย่างการสร้างตารางเส้นทาง

สมมติให้มีเครือข่ายดังรูป 8.2



รูปที่ 8-2 เครือข่ายสาธิตการทำงานของอาร์ไอพี

ในตอนแรกที่เราเตอร์ทั้ง 3 ตัวเริ่มทำงาน ค่าในตารางเส้นทางของเราเตอร์แต่ละตัวที่เกิดจากการติดตั้งค่าของผู้ดูแลเครือข่ายจะเป็นเครือข่ายที่เชื่อมต่อกับเราเตอร์นั้นโดยตรงดังนี้

ตาราง R1

เครือข่าย	เกตเวย์	อินเทอร์เฟส	เมตริก
161.246.10.0	0.0.0.0	e1	1

161.246.15.0	0.0.0.0	e2	1

ตาราง R2

เครือข่าย	เกตเวย์	อินเทอร์เฟส	เมตริก
161.246.15.0	0.0.0.0	e1	1
161.246.25.0	0.0.0.0	e2	1

ตาราง R3

เครือข่าย	เกตเวย์	อินเทอร์เฟส	เมตริก
161.246.25.0	0.0.0.0	e1	1
161.246.30.0	0.0.0.0	e2	1

ตารางเลือกเส้นทางตามตัวอย่างข้างต้นแสดงเพียงฟิลด์สำคัญซึ่งประกอบด้วย

เครือข่าย : ไอพีแอดเดรสของเครือข่ายปลายทาง

เกตเวย์ : ไอพีแอดเดรสของเราเตอร์ซึ่งเป็นทางออกไปสู่เครือข่ายปลายทาง

อินเทอร์เฟส : อินเทอร์เฟสของเราเตอร์

เมตริก : จำนวนขั้น

ในตอนแรกเราเตอร์จะมีตารางเส้นทางเฉพาะเครือข่ายที่อยู่ติดกับเราเตอร์ แต่เส้นทางไปเครือข่ายอื่นนั้นจะได้จากการแลกเปลี่ยนตารางเส้นทางกับเราเตอร์ตัวอื่น ซึ่งอาร์ไอพีนี้กำหนดให้เราเตอร์ประกาศเส้นทางให้กับเราเตอร์ข้างเคียงทุกๆ 30 วินาที หากสมมติให้ R1 บรอดแคสต์ R2 จะได้รับแอดเดรส 161.246.10.0 และ 161.246.15.0 จาก R1

เครือข่าย 161.246.15.0 เป็นค่าที่มีอยู่ในตารางแล้วและเป็นเครือข่ายที่อยู่ติดกับเราเตอร์ R2 จึงไม่เปลี่ยนค่า ส่วนเครือข่าย 161.246.10.0 เป็นค่าใหม่ R2 จะเพิ่มค่าเข้าไปในตารางโดยมองว่า “เราเตอร์ R2 สามารถไปถึงเครือข่าย 161.246.10.0 ได้โดยผ่านเราเตอร์ R1 ซึ่งถ้าเราเตอร์ R1 สามารถไปยังเครือข่าย 161.246.10.0 ได้ด้วยระยะทาง 1 จำนวนขั้น ดังนั้นถ้าเราเตอร์ R2 สามารถไปเครือข่ายนั้นโดยผ่านเราเตอร์ R1 ก็สามารถไปถึงได้ด้วยระยะทางที่เราเตอร์ R1 ไป + 1 จึงเท่ากับ 1+1 = 2 จำนวนขั้น” และใช้เกตเวย์ 161.246.15.1 เป็นทางออก ตาราง ของ R2 หลังจากได้รับตารางเส้นทางที่ R1 บรอดแคสต์ให้จะเป็น

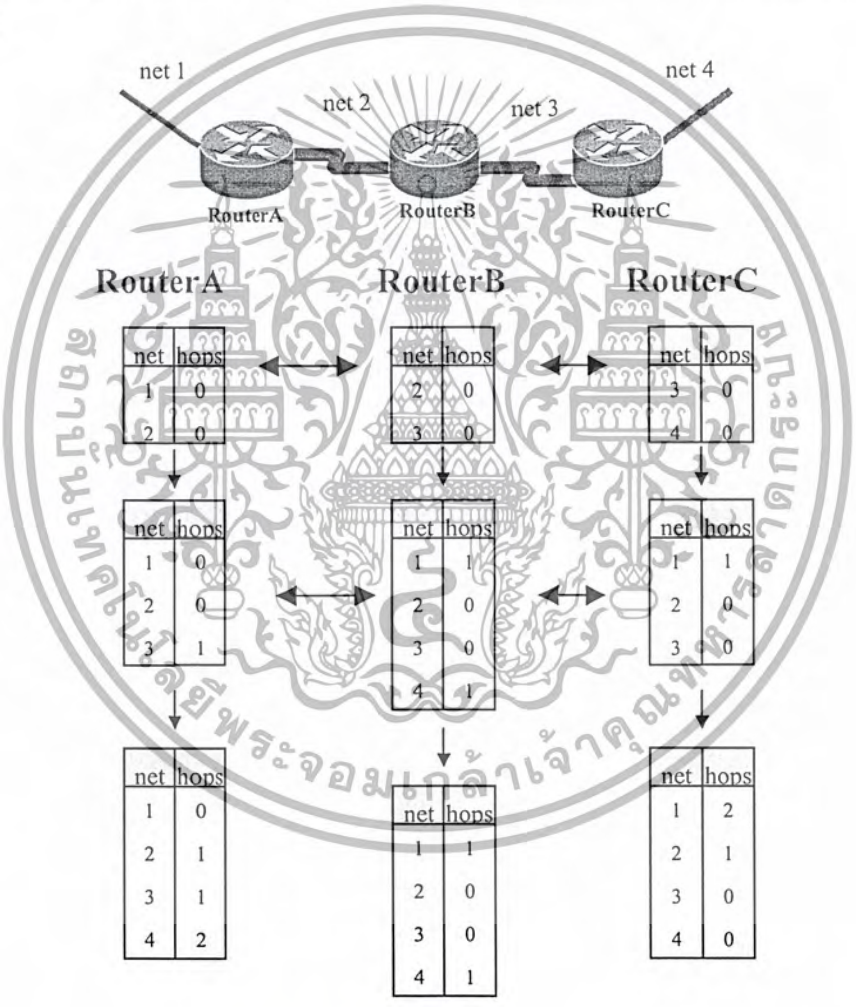
เครือข่าย	เกตเวย์	อินเทอร์เฟส	เมตริก	
161.246.10.0	161.246.15.1	e1	2	← ค่าที่ได้ใหม่
161.246.15.0	0.0.0.0	e1	1	
161.246.25.0	0.0.0.0	e2	1	

ถ้าต่อมา R3 บรอดคาสต์เฉพาะเราเตอร์ R2 ก็จะได้รับค่าและเปลี่ยนค่าในตารางเช่นเดียวกัน ดังนั้นตารางเส้นทางของเราเตอร์ R2 จะเป็น

เครือข่าย	เกตเวย์	อินเทอร์เฟส	เมตริก
161.246.10.0	161.246.15.1	e1	2
161.246.15.0	0.0.0.0	e1	1
161.246.25.0	0.0.0.0	e2	1
161.246.30.0	161.246.25.2	e2	2

← ค่าที่ได้ใหม่

ตารางของ R2 ในตอนแรกมีเพียงแค่เครือข่ายที่เชื่อมต่อโดยตรงกับเราเตอร์นั้น หลังจากที่ได้รับบรอดคาสต์จาก R1 และ R3 ตารางของ R2 ก็จะมีข้อมูลของเครือข่ายครบทุกเครือข่าย รูปที่ 5-3 แสดงการประกาศและปรับค่าตารางเส้นทางของเราเตอร์ทั้งหมด โดยสมมติให้แต่ละเราเตอร์ประกาศค่าตารางพร้อมกัน



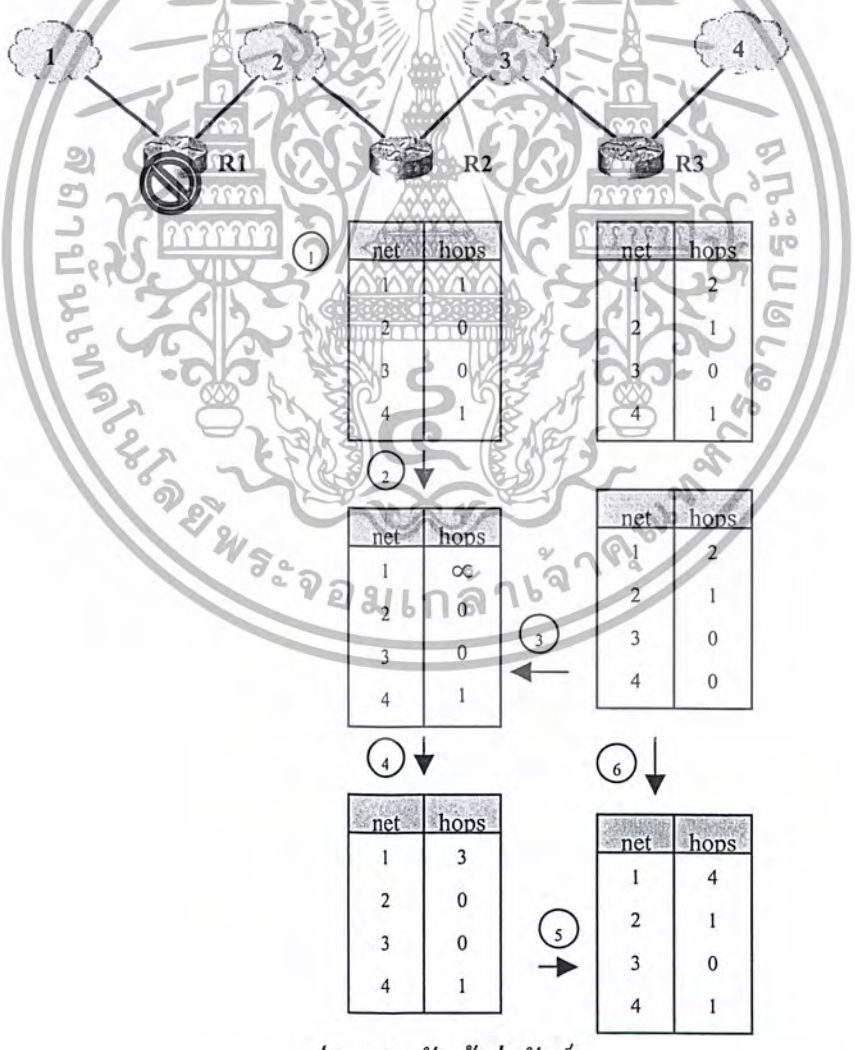
รูปที่ 8-3 การประกาศค่าและปรับค่าและตารางของอาร์ไอพี

8.4.2 การปรับค่าเมื่อเครือข่ายเปลี่ยน

กรณีที่โครงสร้างของเครือข่ายไม่เปลี่ยนแปลง ตารางเส้นทางของอาร์ไอพีก็จะมีเสถียรภาพถึงแม้ว่าเราเตอร์จะประกาศตารางออกไปทุกๆ 30 วินาที จึงเป็นการตรวจสอบสภาพเส้นทางและพร้อมที่จะปรับตารางเส้นทางให้เหมาะสมกับสภาพของเครือข่ายถ้าเส้นทางเดิมนั้นมีปัญหา

เมื่อพิจารณาในทางปฏิบัติแล้วเราเตอร์อาร์ไอพีใดๆ จะไม่สามารถปรับตารางเส้นทางใหม่ได้หากเราเตอร์ข้างเคียงไม่ประกาศค่าเส้นทางให้ ซึ่งอาจเป็นเพราะเราเตอร์ข้างเคียงตัวนั้นไม่สามารถทำงานได้หรือหยุดทำงานชั่วคราว อาร์ไอพีจึงหาวิธีแก้ปัญหานี้โดยการกำหนดให้เส้นทางทุกเส้นทางมีอายุการใช้งานปกติจะกำหนดไว้ที่ 180 วินาที ถ้าเส้นทางใดไม่ได้รับการประกาศค่าเส้นทางใดๆ เป็นเวลาตามที่กำหนดไว้ จะถือว่าเส้นทางนั้นเป็นเส้นทางทีใช้ไม่ได้อีกต่อไป และเราเตอร์จะเปลี่ยนเมตริกของเส้นทางนั้นให้เป็นอนันต์ แต่จะยังไม่ลบเส้นทางนั้นออกจากตาราง หากมีการประกาศเส้นทางนั้นมาจากเราเตอร์อื่น เราเตอร์จะปรับเส้นทางให้ใช้เส้นทางตามที่ได้รับจากเราเตอร์อื่นนั้น

ปัญหาการนับเข้าสู่อนันต์



รูป 8-4 การนับเข้าสู่อนันต์

การทำงานของอาร์ไอพีอาจมีปัญหาเวลาที่เราเตอร์ R พบเส้นทางไปยังเครือข่ายหนึ่งผิดปกติ ในขณะเดียวกันเราเตอร์ข้างเคียงตัวหนึ่งแจ้งว่ามีเส้นทางไปยังเครือข่ายที่เกิดผิดปกติ เราเตอร์ R ที่พบปัญหาก็จะปรับค่าเส้นทางตามค่าที่ได้รับมาใหม่ แต่เนื่องจากเป็นเส้นทางที่เราเตอร์ข้างเคียงสามารถไปถึงเครือข่ายนั้นได้โดยผ่านทางเราเตอร์ R ทำให้เกิดการปรับเมตริกต่อไปแบบไม่มีที่สิ้นสุด ปรากฏการณ์นี้เราเรียกว่า การนับเข้าสู่อนันต์ (Count to Infinity)

ให้ตารางเส้นทางที่เราเตอร์ R2 และ R3 มีค่าดังตำแหน่ง ในจังหวะที่เราเตอร์ R1 ทำงานผิดปกติ ทำให้เครือข่าย 1 จะถูกตัดขาดออกไป ค่าเส้นทางของเครือข่าย 1 ในR2 จะลดลงตามการทำงานจองตัวจับเวลา เมื่อเมื่อ 180 วินาที เส้นทางไปเครือข่าย 1 จะเปลี่ยนเป็นอนันต์ดังตำแหน่ง

หากเวลานั้นถึงเวลาที่ R3 บรอดแคสต์ R3 จะส่งค่าเส้นทางมาให้ R2 ทำให้ R2 ได้รับเส้นทางไปเครือข่าย 1 มาด้วยเมตริก 2 ดังนั้น R2 จึงเปลี่นค่าในตารางเส้นทางให้เส้นทางไปเครือข่าย 1 สามารถไปโดยผ่านทาง R3 ด้วยเมตริก 3 แต่ที่จริงแล้วเครือข่าย 1 นั้นไม่สามารถใช้งานได้อยู่ในขณะนั้น R2 และ R3 ก็ไม่ทราบได้โดยตรง ทำให้เมื่อถึงเวลา R2 และ R3 ประกาศตารางเส้นทางตารางของทั้งสองจะปรับค่าเส้นทางเพื่อไปยังเครือข่าย 1 อย่างไม่มีที่สิ้นสุด ซึ่งอาร์ไอพีได้กำหนดค่าเมตริกไว้ค่าหนึ่งเพื่อแก้ปัญหานี้ ค่าเมตริกนี้ควรเป็นค่าที่ไม่สูงมากเพราะจะได้ไม่ต้องใช้เวลานาน แต่ถ้าน้อยเกินไปก็อาจทำให้เครือข่ายถูกจำกัดให้เล็กลง ซึ่งปกติจะกำหนดไว้ที่ 16 เมื่อเราเตอร์เพิ่มค่าไปถึง 16 ก็จะทราบว่าเส้นทางนั้นเป็นเส้นทางที่ไม่สามารถไปถึงได้ และจะเข้าสู่กระบวนการจับเวลาเพื่อกำจัดเส้นทางออกจากตาราง

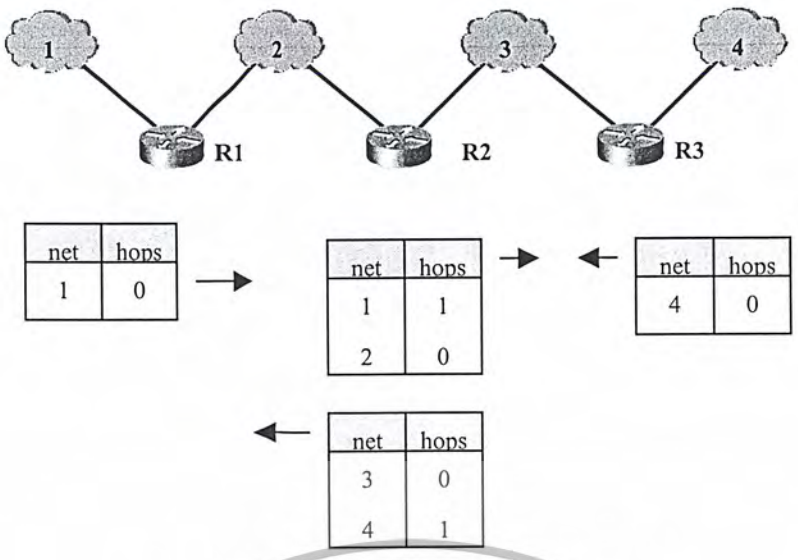
เวลาลู่เข้า

เวลาลู่เข้าของอาร์ไอพีหมายถึงเวลาที่เราเตอร์ใช้ปรับตารางจนกระทั่งตารางมีค่าถูกต้องตามโครงสร้างของเครือข่ายจริง จากข้อกำหนดของการประกาศค่าของเราเตอร์อาร์ไอพีทุกๆ 30 วินาที ปัญหาการนับเข้าสู่อนันต์ในเครือข่ายที่ใช้อาร์ไอพีจะใช้เวลาสูงสุดประมาณ 7 นาที

หากเวลาลู่เข้าของอาร์ไอพีสั้นลง เครือข่ายก็ย่อมที่จะเข้าสู่เสถียรภาพได้เร็วขึ้น วิธีปรับเวลาลู่เข้าของอาร์ไอพีให้สั้นลงมีหลายวิธี ดังนี้

8.4.2.1 สปลิตฮอไรซัน(Split Horizon)

วิธีนี้จะช่วยแก้ปัญหานับเข้าสู่อนันต์ (เฉพาะเครือข่ายบางรูปแบบเท่านั้น) โดยเราเตอร์จะไม่ประกาศตารางเส้นทางทั้งตารางให้กับเราเตอร์ข้างเคียงแต่จะประกาศเฉพาะเส้นทางที่ไม่ได้เรียนรู้มาจากเราเตอร์ตัวนั้นเท่านั้น เช่นเราเตอร์ R2 ได้รับตารางเส้นทางไปเครือข่าย 1 มาจากเราเตอร์ R1 เราเตอร์ R2 ก็จะประกาศเฉพาะเส้นทางของเครือข่าย 3 และ 4 ไปให้กับ R1 ดังรูป



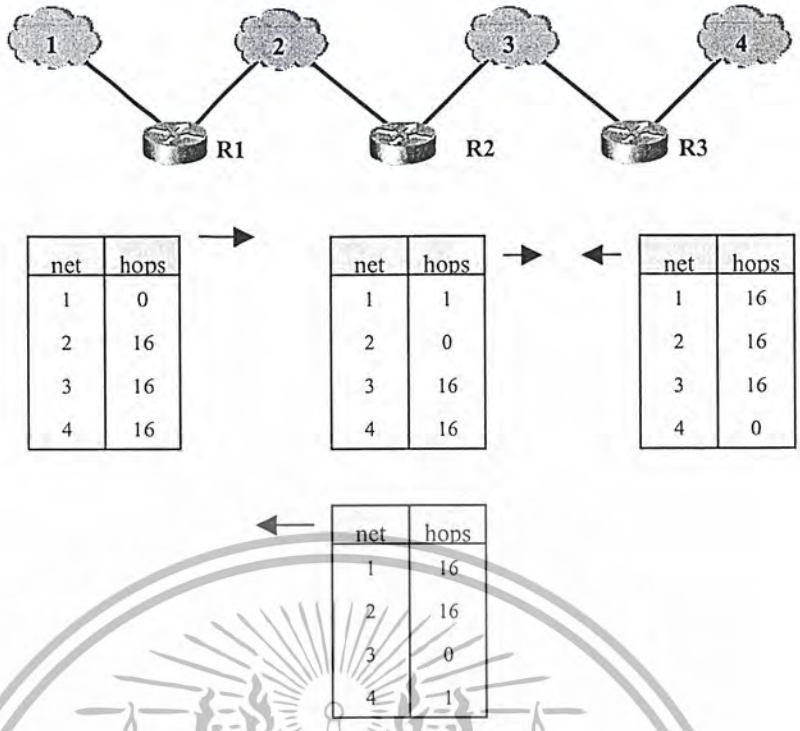
รูปที่ 8-5 การประกาศเส้นทางด้วยวิธีสปลิตฮอไรซัน

หากเราเตอร์ R1 หยุดทำงานไป ค่าเส้นทางของเครือข่าย 1 ใน R2 จะค่อยๆ ลดลงจนครบ 180 วินาที เส้นทางไปเครือข่าย 1 จะเปลี่ยนเป็นอนันต์ โดยที่ R2 จะไม่ได้รับการประกาศเส้นทางไปเครือข่าย 1 จาก R3 เนื่องจากเป็นเส้นทางที่ R2 ประกาศให้ R3 รู้วิธีไปยังเครือข่าย 1 เอง ดังนั้นจึงไม่เกิดปัญหานับเข้าสู่อนันต์

8.4.2.2 สปลิตฮอไรซันแบบพอยซันรีเวอร์ส (Split Horizon with Poisoned Reverse)

วิธีนี้เป็นเทคนิคอีกแบบหนึ่งของสปลิตฮอไรซันที่ใช้วิธีแจ้งค่าเส้นทางที่ชัดเจนมากยิ่งขึ้น เพื่อไม่ให้เกิดการนับเข้าสู่อนันต์

วิธีนี้จะยอมให้เราเตอร์ประกาศเส้นทางที่เรียนรู้จากลิงค์หนึ่งๆ กลับไปได้ แต่ให้เมตริกประจำเส้นทางนั้นเป็นอนันต์หรือเท่ากับ 16 เพื่อกำกับว่าเส้นทางนั้นไม่สามารถใช้งานได้ เช่น มีเราเตอร์ต่อกันดังรูป



รูปที่ 8-6 การประกาศเส้นทางด้วยวิธีสปลิตฮอไรซันแบบพอยซันรีเวอร์ส

หากเราเตอร์ R1 หยุดทำงาน ค่าเส้นทางของเครือข่าย 1 ใน R2 จะลดลงตามเวลาการทำงานของตัวจับเวลา และเมื่อครบ 180 วินาที เส้นทางไปเครือข่าย 1 จะเปลี่ยนเป็นอนันต์ เมื่อทั้ง R2 และ R3 ต่างประกาศค่าซึ่งแจ้งว่าระยะทางไปเครือข่าย 1 มีค่าเท่ากับ 16 สวนทางกันก็จะทราบได้ว่าเครือข่าย 1 ใช้งานไม่ได้ ทั้ง R2 และ R3 ก็จะกำจัดเส้นทางไปเครือข่าย 1 โดยไม่เกิดปัญหานับเข้าสู่อนันต์

วิธีนี้จะสร้างแพ็กเก็ตมากกว่าวิธีสปลิตฮอไรซันแบบธรรมดา เนื่องจากมีค่าที่ต้องประกาศมากกว่า แต่ก็เป็นวิธีที่เราเตอร์ในปัจจุบันนิยมใช้เนื่องจากให้ประสิทธิผลดีกว่า

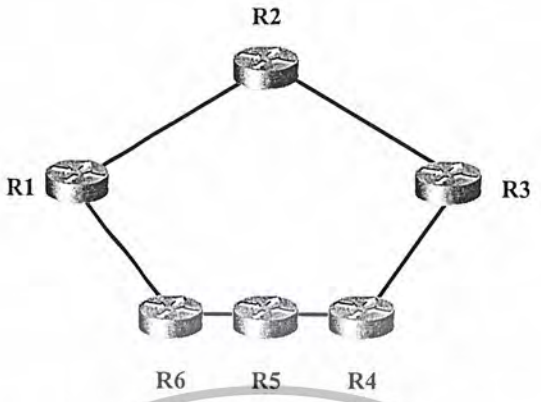
เทคนิคของสปลิตฮอไรซันทั้งสองแบบนี้ไม่สามารถป้องกันปัญหานับเข้าสู่อนันต์ได้ทุกรูปแบบเครือข่าย ช่วยแก้ปัญหาสำหรับเราเตอร์สองตัวที่อยู่ข้างเคียงกันเท่านั้น

8.4.2.3 ทริกเกอร์อัพเดต

สปลิตฮอไรซันทั้ง 2 วิธีนั้นช่วยแก้ปัญหาการนับเวลาเข้าสู่อนันต์เพื่อลดเวลาลู่เข้า หากแต่ยังต้องใช้เวลาครู่หนึ่งก่อนที่จะตรวจสอบว่าเส้นทางหนึ่งๆ มีปัญหา ทริกเกอร์อัพเดตเป็นเทคนิคเสริมที่ช่วยลดเวลาลู่เข้าของอาร์ไอพี คือกำหนดให้เราเตอร์ประกาศค่าเส้นทางออกไปทันทีที่พบว่าโครงสร้างเครือข่ายมีการเปลี่ยนแปลงโดยไม่ต้องรอให้ครบเวลา 30 วินาที

หากเราเตอร์ทุกตัวใช้ทริกเกอร์อัพเดตก็จะสามารถปรับปรุงตารางได้อย่างรวดเร็ว แต่ก็อาจจะสร้างภาระในเครือข่ายขนาดใหญ่ที่มีเราเตอร์หลายตัว เพราะการเปลี่ยนแปลงใดๆ ก็ย่อมทำให้เราเตอร์ตัวหนึ่งส่งข้อมูลให้เราเตอร์ตัวอื่นประกาศค่าตามกันไปอย่างต่อเนื่องหรือเกิดแพ็กเก็ตบรอดแคสต์เป็นจำนวนมาก

แต่เทคนิคนี้อาจทำให้เกิดเหตุการณ์ที่ไม่คาดฝันขึ้นได้ เช่นในกรณีที่มีเครือข่ายดังรูป



รูปที่ 8-7 เครือข่ายที่จำเป็นต้องใช้โฮลดาวน์ร่วมกับทริกเกอร์อัพเดต

ถ้าเส้นทางที่เชื่อมระหว่าง R3 กับ R4 ไม่สามารถใช้งานได้และใช้เทคนิคทริกเกอร์อัพเดตแจ้งไปยังเราเตอร์ที่อยู่ข้างเคียง ปัญหาจะเกิดขึ้นเมื่อ R1 ได้รับการแจ้งค่าจาก R2 ก่อน R6 ทำให้ R1 ปรับตารางว่าเส้นทางที่เชื่อมระหว่าง R3 กับ R4 ใช้งานไม่ได้

ต่อมาก่อนที่ R6 จะได้รับแจ้งจาก R5 ถ้าถึงเวลาประกาศเส้นทางออกไป ทำให้ R1 ได้รับแจ้งว่ามีเส้นทางระหว่าง R3 กับ R4 ใช้งานได้ผ่านไปทาง R6 ทำให้ R1 ปรับตารางเส้นทางกลับไปใหม่ว่าเส้นทางระหว่าง R3 กับ R4 ใช้งานได้ ทำให้กว่าที่เราเตอร์ทุกตัวจะรู้ว่าเส้นทางระหว่าง R3 กับ R4 ไม่สามารถใช้งานได้ ก็ใช้เวลานานพอสมควร เพื่อแก้ปัญหานี้จึงใช้กฎเพิ่มเติมสำหรับการปรับตารางที่เรียกว่าโฮลดาวน์ (Hold Down)

โฮลดาวน์กำหนดช่วงเวลาให้เราเตอร์งดปรับเปลี่ยนค่าที่เพิ่งจะตรวจพบว่าไม่สามารถใช้งานได้ จากตัวอย่างข้างต้นนี้ เมื่อ R1 ปรับเส้นทางระหว่าง R3 กับ R4 ว่าไม่สามารถใช้งานได้กฎโฮลดาวน์จะบังคับให้ R1 รอเวลาโดยไม่รับการแจ้งค่าว่าเส้นทางระหว่าง R3 กับ R4 สามารถใช้งานได้จากเราเตอร์อื่นเป็นเวลาหนึ่ง โดยช่วงเวลานี้ควรเป็นช่วงเวลาที่มากพอจะให้ทริกเกอร์อัพเดตกระจายไปยังเราเตอร์ทุกตัว ซึ่งการใช้โฮลดาวน์ถึงแม้ว่าจะทำให้ช่วงเวลาลู่เข้านานกว่าแบบไม่ใช้ แต่ทำให้การทำงานของระบบดีกว่าแบบไม่ใช้ด้วย

8.4.3 ตัวจับเวลาในอาร์ไอพี

ตัวจับเวลาในอาร์ไอพีจะมีอยู่ 3 ประเภทคือ

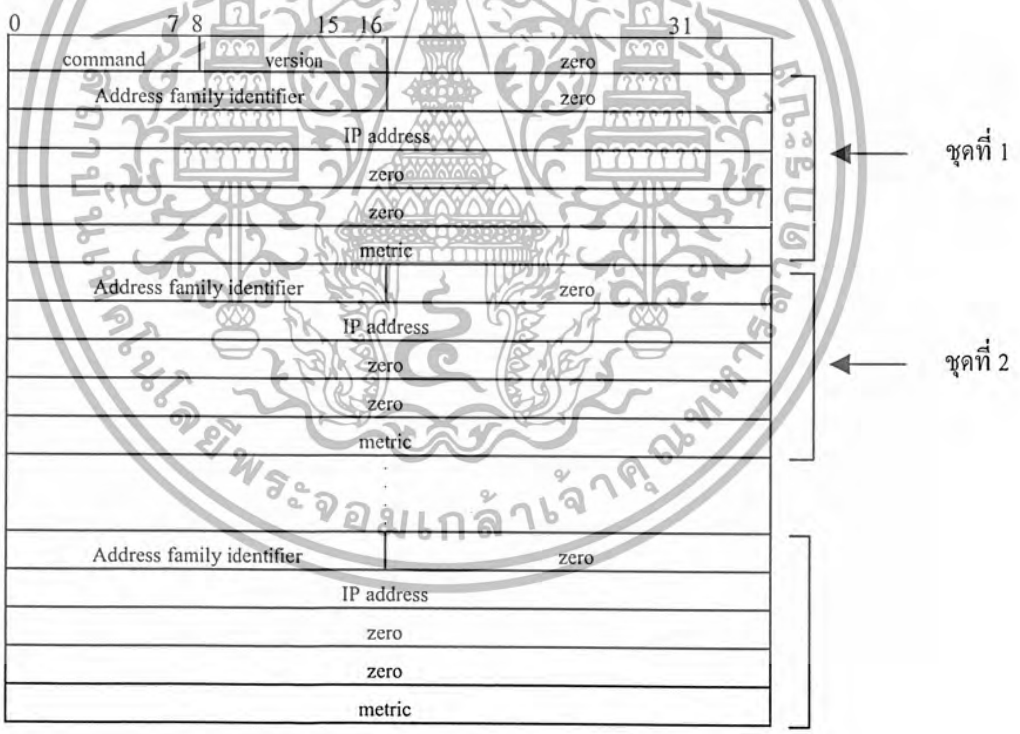
1. **ตัวจับเวลาปรับค่า (Update Timer)** เป็นช่วงเวลาที่กำหนดการบรอดคาสต์ของเราเตอร์ ซึ่งปกติจะให้บรอดคาสต์ตารางทุกๆ 30 วินาที แต่ในทางปฏิบัติเราไม่สามารถหลีกเลี่ยงไม่ให้เกิดการบรอดแคสต์พร้อมกัน เช่น ในเราเตอร์ซิสโก้จะสุ่มค่า RIP_JITTER ซึ่งมีค่าได้ไม่เกิน 4.5 วินาทีแล้วนำมาลบกับค่า 30 ตัวจับเวลาปรับค่าของเราเตอร์ซิสโก้จึงอยู่ระหว่าง 25.5 ถึง 30 วินาที

วิธีนี้จึงช่วยลดโอกาสเกิดการบรอดแคสต์แพ็กเก็ตพร้อมกันเป็นจำนวนมากลงไป นอกจากนี้เพื่อป้องกันการกลับมาเข้าจังหวะของเราเตอร์เมื่อเกิดทริกเกอร์อัพเดต เราเตอร์จะต้องไม่รีเซ็ตตัวจับเวลาให้กลับมาเป็นศูนย์เมื่อเกิดทริกเกอร์อัพเดต แต่ให้จับเวลาต่อไปตามปกติและบรอดคาสต์ออกไปเมื่อถึงเวลา

2. **ตัวจับเวลาหมดอายุ (Expiratin Timer)** เป็นตัวกำหนดอายุของแต่ละเส้นทาง โดยถ้าเส้นทางใดไม่ได้รับประกาศเส้นทางนั้นมาจากเราเตอร์อื่นเป็นเวลาเท่าที่กำหนดนี้ ค่าเมตริกของเส้นทางนั้นก็จะถูกเปลี่ยนเป็น 16 เพื่อแสดงว่าไม่สามารถไปถึงได้ โดยปกติจะกำหนดเวลานี้เป็น 180 วินาที
3. **ตัวจับเวลากำจัดเส้นทาง (Garbage Collection Timer)** สำหรับเส้นทางที่หมดอายุจากการจับเวลา 180 วินาทีข้างต้นแล้วจะยังไม่ถูกลบออกจากตารางทันที แต่จะนับเวลาถอยหลังเป็นจำนวนเวลาตามที่กำหนดนี้เพื่อลบเส้นทางนี้ออกไปจากตาราง แต่ในระหว่างที่นับเวลาถอยหลังนี้ก็จะบรอดแคสต์เส้นทางนี้ออกไปด้วย เพื่อให้เราเตอร์ตัวอื่นนับเวลาถอยหลังเพื่อลบเส้นทางนี้ด้วย

8.4.4 เฟรมอาร์ไอพี

เฟรมอาร์ไอพีจะนำส่งโดยบรรจุอยู่ในยูดีพี ลักษณะของเฟรมอาร์ไอพีจะเป็นดังรูป



รูปที่ 8-8 ฟอร์แมตของเฟรมอาร์ไอพี

รูปแบบเฟรมในที่นี้แสดงเฉพาะการใช้อาร์ไอพีในเครือข่ายไอพีเท่านั้น หากเป็นเครือข่ายอื่นจะมีรูปแบบแตกต่างออกไป โดยแต่ละฟิลด์มีความหมายดังนี้

- command ขนาด 8 บิต : กำหนดแบบการทำงานว่าเป็นการร้องขอ (ค่าเท่ากับ 1) หรือการตอบรับ (ค่าเท่ากับ 2) การบรอดคาสต์ของอาร์ไอพีจัดเป็นการตอบรับ (เพราะไม่มีการร้องขอมาก่อน) อาร์ไอพีจะกำหนดคำสั่งให้เป็น 1 เพื่อร้องขอต่อเมื่อต้องการทราบตารางเส้นทางของเราเตอร์ใดเป็นกรณีเฉพาะ
- version ขนาด 8 บิต : กำหนดรุ่นของโพรโตคอล หากเท่ากับ 1 หมายถึงรุ่น 1
- zero ขนาด 16 บิต : สงวนไว้และต้องมีค่าเป็น 0
- address family indentier ขนาด 16 บิต : การออกแบบอาร์ไอพีในขั้นต้นไม่ได้กำหนดให้ใช้เฉพาะกับทีซีพี/ไอพีเท่านั้น จึงมีฟิลด์กำกับชุดแอดเดรสเพื่อแยกแยะชนิดโพรโตคอลที่อาร์ไอพีทำงานด้วย ค่านี้เท่ากับ 2 สำหรับไอพี
- IP address ขนาด 32 บิต : ไอพีแอดเดรสของเครือข่ายที่ประกาศค่าออกไป
- Metric ขนาด 32 บิต : เมตริกกำหนดระยะทางให้ใช้ค่าได้ตั้งแต่ 1 ถึง 15 หากมีค่าเป็น 16 หมายถึงเครือข่ายปลายทางไม่สามารถไปถึงได้

ตั้งแต่ฟิลด์ address family identifier กระทั่งถึงเมตริกสามารถมีค่าซ้ำได้ 25 ชุด เพื่อให้เราเตอร์สามารถบรอดคาสต์เส้นทางได้ 25 ค่าในคาต้าแกรมเดียว

เฟรมอาร์ไอพี 2

อาร์ไอพีรุ่น 2 มีส่วนขยายการทำงานเพิ่มเติมจากอาร์ไอพีรุ่นแรกและเพิ่มระบบความปลอดภัยในการส่งข้อมูลเส้นทางระหว่างเราเตอร์ อาร์ไอพีรุ่น 1 ไม่สนับสนุนการใช้งานในเครือข่ายที่มีซับเน็ตแบบแปรค่า ไม่สามารถแจ้งหมายเลขระบบออโตโนมัสเพื่อเชื่อมความปลอดภัยในการส่งตาราง อาร์ไอพีรุ่น 2 จึงได้รับการพัฒนาขึ้นเพื่อรักษาความปลอดภัยในการส่งตาราง อาร์ไอพีรุ่น 2 จึงได้รับการพัฒนาขึ้นเพื่อแก้ปัญหาดังกล่าวและกลายเป็นมาตรฐานใหม่ทดแทนอาร์ไอพีรุ่น 1

ฟิลด์อาร์ไอพีรุ่น 2 ยังคงลักษณะของอาร์ไอพีรุ่น 1 แต่ได้เพิ่มเติมฟิลด์ใหม่ และมีรูปแบบดังรูป 5.10

ความหมายของแต่ละฟิลด์

Command ขนาด 8 บิต : กำหนดแบบการทำงานว่าเป็นการร้องขอ (ค่าเท่ากับ 1) หรือเป็นการตอบรับ (ค่าเท่ากับ 2)

Version ขนาด 16 บิต : กำหนดรุ่นของโพรโตคอล ค่าที่ใช้คือ 2

Unused ขนาด 16 บิต : ไม่ได้ใช้งาน

Address family identifier ขนาด 16 บิต : ฟิลด์กำกับชุดแอดเดรสเพื่อแยกแยะชนิดของโพรโตคอลที่อาร์ไอพีทำงานร่วมด้วย สำหรับไอพีใช้ค่าเท่ากับ 2

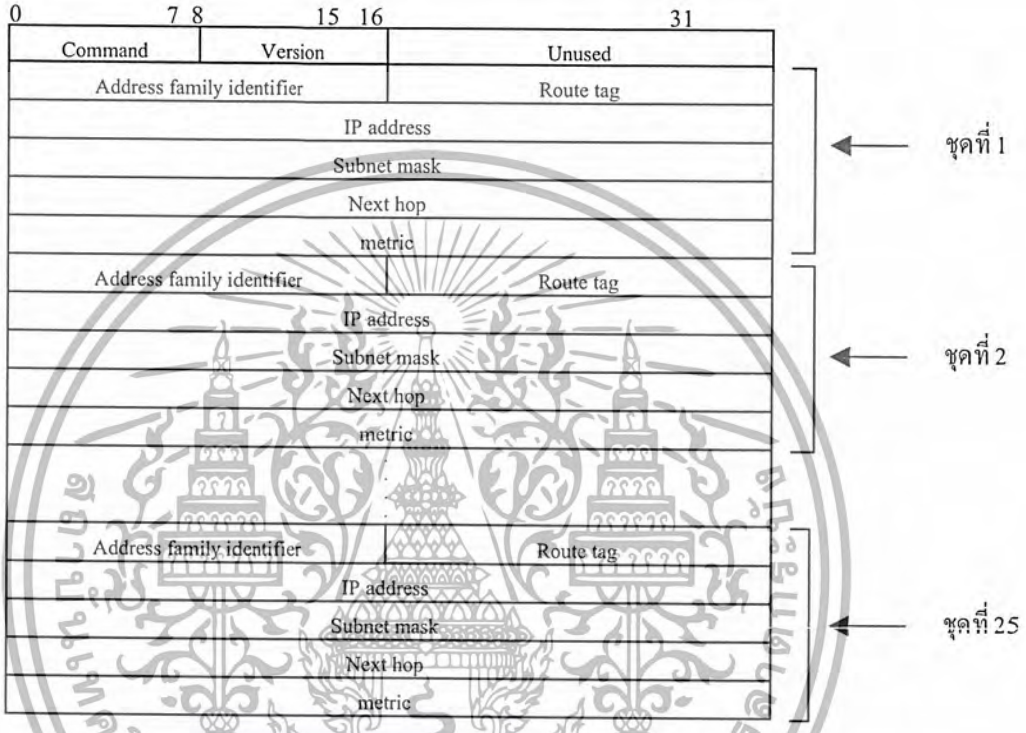
Route tag ขนาด 8 บิต : ใช้แยกความแตกต่างระหว่างการได้มาของค่าเส้นทางว่าเรียนรู้จากภายในหรือภายนอก ค่าที่ใช้อาจเป็นเลขระบบออโตโนมัสสำหรับเส้นทางที่ได้มาจากโพรโตคอลภายนอก

IP Address ขนาด 32 บิต : ไอพีแอดเดรสที่กำหนดเส้นทาง

Subnet Mask ขนาด 32 บิต : ซับเน็ตมาสก์กำกับไอพีแอดเดรส หากมีค่าเป็น 0 หมายถึงไม่ได้กำหนดซับเน็ตมาสก์ให้

Next Hop ขนาด 32 บิต : ไอพีแอดเดรสของเราเตอร์ถัดไปที่ควรจะนำส่งแพ็กเก็ตซึ่งมีแอดเดรสปลายทางตามที่กำหนดด้วยค่าเลือกเส้นทางในเฟรมอาร์ไอพีนี้ ประโยชน์ของฟิลด์นี้ใช้เพื่อลดจำนวนเส้นทางที่แพ็กเก็ตต้องเดินทางผ่านในกรณีที่มีโพรโตคอลเลือกเส้นทางอื่นนอกเหนือไปจากอาร์ไอพีทำงานร่วมกัน หากมีค่า 0.0.0.0 หมายถึงให้ส่งแพ็กเก็ตไปยังเราเตอร์ที่ประกาศค่าเส้นทางนี้มาให้

Metric ขนาด 32 บิต : เมตริกกำหนดระยะทางยังคงมีค่าได้ตั้งแต่ 1 ถึง 15 เช่นเดียวกับอาร์ไอพีรุ่น 1การพิสูจน์ตัวจริง

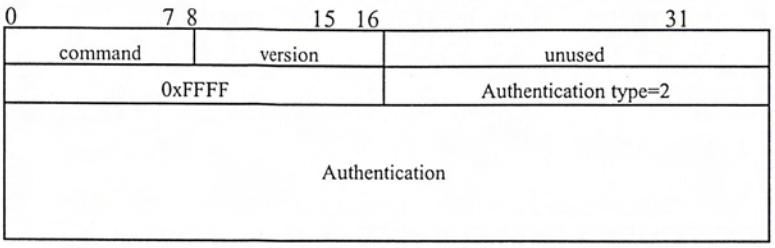


รูปที่ 8-9 ฟอร์แมตของเฟรมอาร์ไอพีเวอร์ชัน 2

ตั้งแต่ฟิลด์ address family identifier ถึงฟิลด์ metric มีค่าซ้ำกันได้ 25 ชุดเช่นเดียวกับเวอร์ชันแรก เพื่อการประกาศเส้นทางไปพร้อมกันได้ 25 เส้นทางในเฟรมเดียว แต่ถ้าฟิลด์ address family identifier ชุดแรกมีค่าเป็น 0xFFFF (ให้ค่านี้มีได้เฉพาะในชุดแรกเท่านั้น) หมายความว่าเป็นเฟรมกำหนดการพิสูจน์ตัวจริง และฟิลด์ถัดมาจะถูกแปลความหมายใหม่ตามรูป 5-10

มีฟิลด์ authentication type ขนาด 16 บิตอยู่ต่อจาก address family identifier และถัดไปอีก 128 บิตเป็นฟิลด์ authentication สำหรับใช้ตรวจสอบว่าเราเตอร์ที่ส่งเฟรมนี้เป็นเราเตอร์ตัวจริงที่ได้รับอนุญาต

ในปัจจุบันกำหนดให้ฟิลด์ authentication type มีค่าเท่ากับ 2 ซึ่งหมายถึงการพิสูจน์ด้วยรหัสผ่าน (ชนิดที่ไม่เข้ารหัสลับ) และค่ารหัสผ่านนั้นจะต้องส่งมาในฟิลด์ authentication ด้วยเหตุนี้เฟรมอาร์ไอพีที่ใช้การพิสูจน์ตัวจริงจะมีจำนวนเส้นทางที่บรอดคาสต์พร้อมกันในหนึ่งเฟรมได้เพียง 24 เส้นทาง



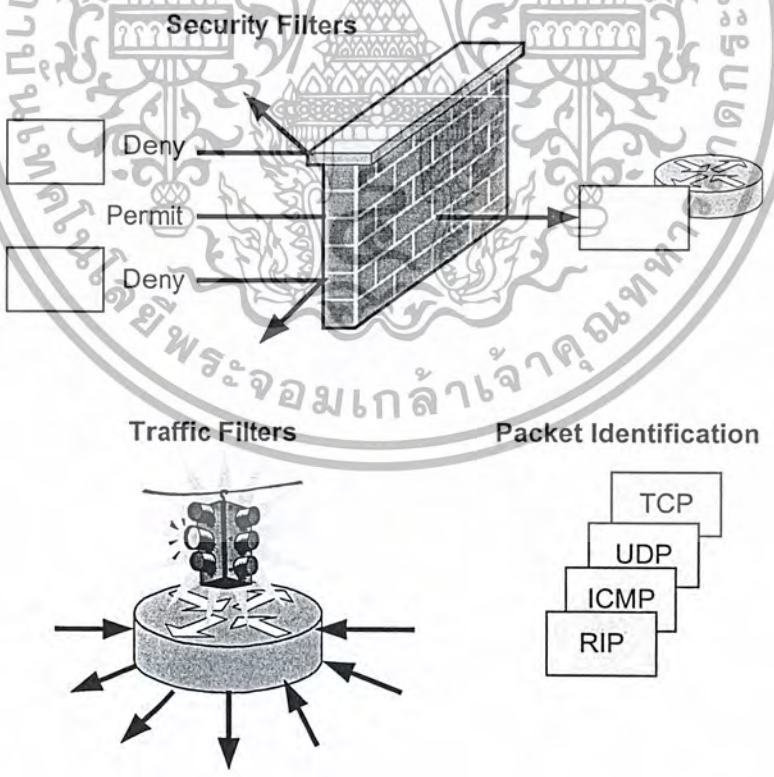
รูปที่ 8-10 เฟรมอาร์ไอพีเวอร์ชัน 2 เมื่อใช้การพิสูจน์ตัวจริง

บทที่ 9

แอคเซสลิสต์(Access list)

แอคเซสลิสต์เป็นวิธีการที่รู้จักกันโดยทั่วไปในปัจจุบันซึ่งการทำงานประกอบด้วยการอนุญาตและการป้องกันของแพ็กเก็ตที่เข้าหรือออกจากเราเตอร์ ซึ่งแอคเซสลิสต์กลายเป็นเครื่องมือสำหรับใช้ควบคุมแพ็กเก็ตและภายในเครือข่ายที่มีประสิทธิภาพในปัจจุบัน ซึ่งมีรูปแบบการทำงานหลัก 3 แบบดังนี้ คือ

1. การตรวจสอบความปลอดภัย จะทำการอนุญาตเฉพาะแพ็กเก็ตที่รู้จักและทำการป้องกันแพ็กเก็ตที่เหลือออกไปทั้งหมด
2. การตรวจสอบการจารจรในเครือข่ายจะทำการป้องกันแพ็กเก็ตที่มีความสำคัญน้อยหรือไม่มีความสำคัญออกไป ซึ่งจะทำให้ไม่เสียขนาดแบนด์วิดธ์ในเครือข่ายไป วิธีการนี้คล้ายกับ การตรวจสอบความปลอดภัย แต่จะใช้เทคนิคการใช้งานตรงข้ามกัน โดยจะทำการป้องกันแพ็กเก็ตที่ไม่ต้องการออก และจะอนุญาตแพ็กเก็ตที่เหลือให้อยู่ต่อไป
3. เป็นเครื่องมือที่ใช้ในเราเตอร์ของซิสโก้ ยกตัวอย่างเช่น ไดอัลเลอร์ลิส เราเตอร์ฟิวล์เตอร์ เราแม็บ และ คิวริ่งลิส ซึ่งสามารถบ่งชี้ได้ว่ารูปแบบแพ็กเก็ตเป็นแบบใด

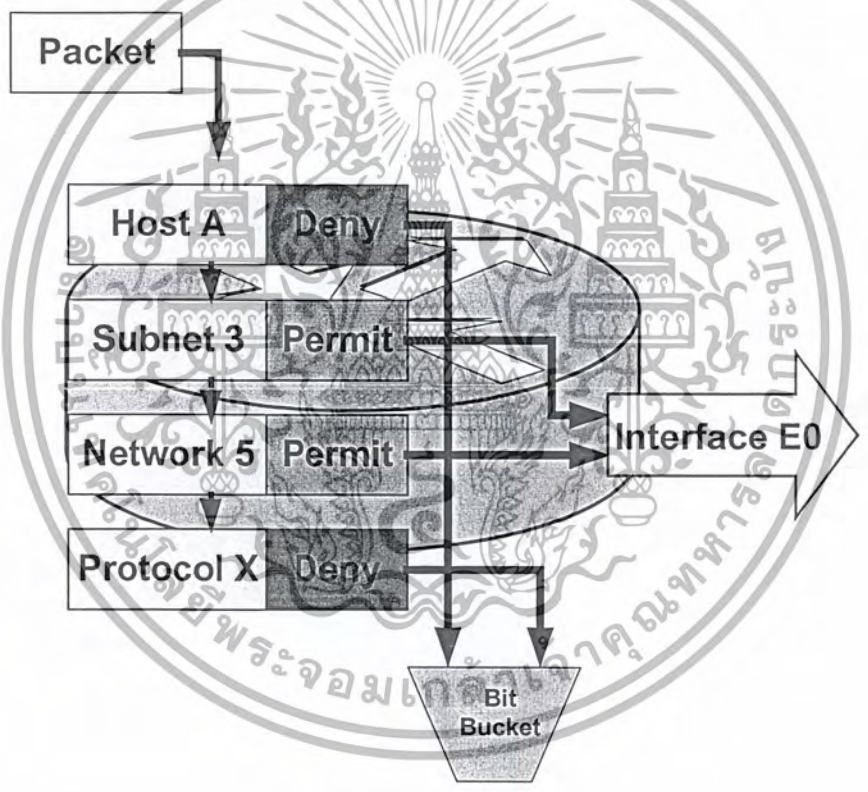


รูปที่ 9-1 การนำแอคเซสลิสต์ไปใช้งาน

9.1 พื้นฐานของแอคเซสลิสต์

แอคเซสลิสต์เป็นวิธีการตรวจสอบแบบเป็นลำดับและต่อเนื่องโดยการทดสอบจะเปรียบเทียบจาก แพ็กเก็ตข้อมูลที่เข้ามา และจะเลือกว่าจะกระทำกับชุดข้อมูลนั้นอย่างไร ระหว่างการอนุญาตหรือป้องกัน ในส่วนการเปรียบเทียบจะทำได้ตั้งแต่การเปรียบเทียบแอดแดรสต้นทาง หรืออาจมีความซับซ้อนยิ่งขึ้น เช่น การตรวจสอบทั้งแอดแดรสต้นทางและแอดแดรสปลายทาง รูปแบบทางโพรโตคอล หมายเลขพอร์ต เป็นต้น

แพ็กเก็ตจะถูกนำเข้ามาไว้ที่บนสุดของ สแตกฟิวล์เตอร์ ดังรูปที่2 และในแต่ละขั้นตอนการตรวจเช็คถ้าตรวจเช็คแล้วเจอเงื่อนไขที่ตรงกันก็จะไปทำต่อในการทำงานที่เลือกไว้โดยอาจเป็นการอนุญาตหรือการป้องกันแพ็กเก็ตนั้นแต่ถ้าการเปรียบเทียบไม่ตรงกับเงื่อนไขแพ็กเก็ตนั้นจะถูกส่งไปเพื่อการเปรียบเทียบในขั้นตอนอื่นต่อ และการเปรียบเทียบก็จะเกิดขึ้นต่อไป

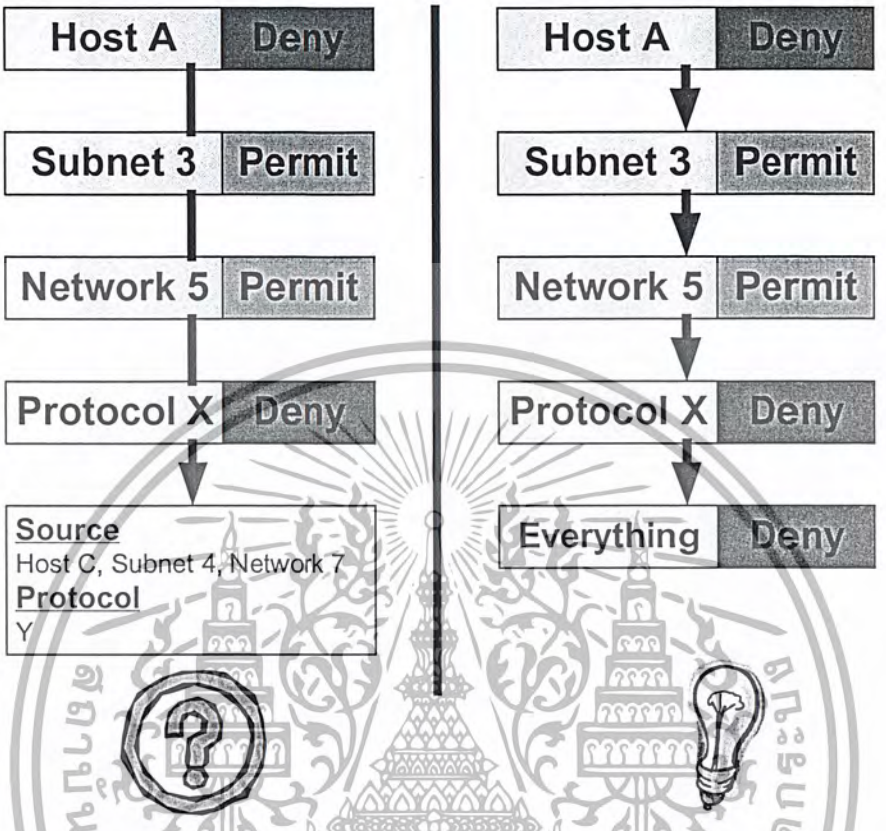


รูปที่ 9-2 ลำดับการทำงานของแอคเซสลิสต์

ในรูปที่ 9-2 คำว่า *Permit* หมายถึงแพ็กเก็ตได้รับอนุญาตให้ออกไปจาก อินเทอร์เฟสอีเทอร์เน็ตหมายเลข 0 (E0) และ *Deny* จะหมายถึงแพ็กเก็ตจะถูกกำจัดออกไป ตามตัวอย่างในรูป สมมุติให้แพ็กเก็ตมีรูปแบบดังนี้

Host D - Subnet 2 - Network 5 แล้วในการเทียบครั้งแรกซึ่งมีเงื่อนไขเป็น Host A จะไม่ตรงซึ่งจะทำให้แพ็กเก็ตถูกส่งไปยังข้างด้านล่างต่อมาเพื่อเปรียบเทียบกับ Subnet 3 ซึ่งจะยังไม่ตรงอีก สุดท้ายเมื่อเปรียบ

เทียบกับ Network 5 จะทำให้แพ็กเก็ตนั้นได้รับการส่งออกไปจาก อินเทอร์เฟสอีเทอร์เน็ตหมายเลข 0 ในที่สุด



รูปที่ 9-3 ตัวอย่างการจัดการกับแพ็กเก็ตที่ไม่ตรงในแต่ละลำดับขั้น

ส่วนในกรณีที่การเปรียบเทียบในทุกขั้นตอนแล้วไม่ตรงการเปรียบเทียบใดๆ จะใช้ค่าสำรองที่จะมีอยู่แล้วในที่นี้ใน เราเตอร์ของ ซิสโก้ จะเป็น *Deny Any* หรือหมายความว่าให้ทำการกำจัดแพ็กเก็ตที่เหลือออกไป เราสามารถทำกรเปลี่ยนค่าสำรองนี้ได้โดยอาจกำหนด ค่าให้เป็น *Permit Any* เพื่อทำการอนุญาตแพ็กเก็ตที่เหลือจากการเปรียบเทียบทั้งหมด

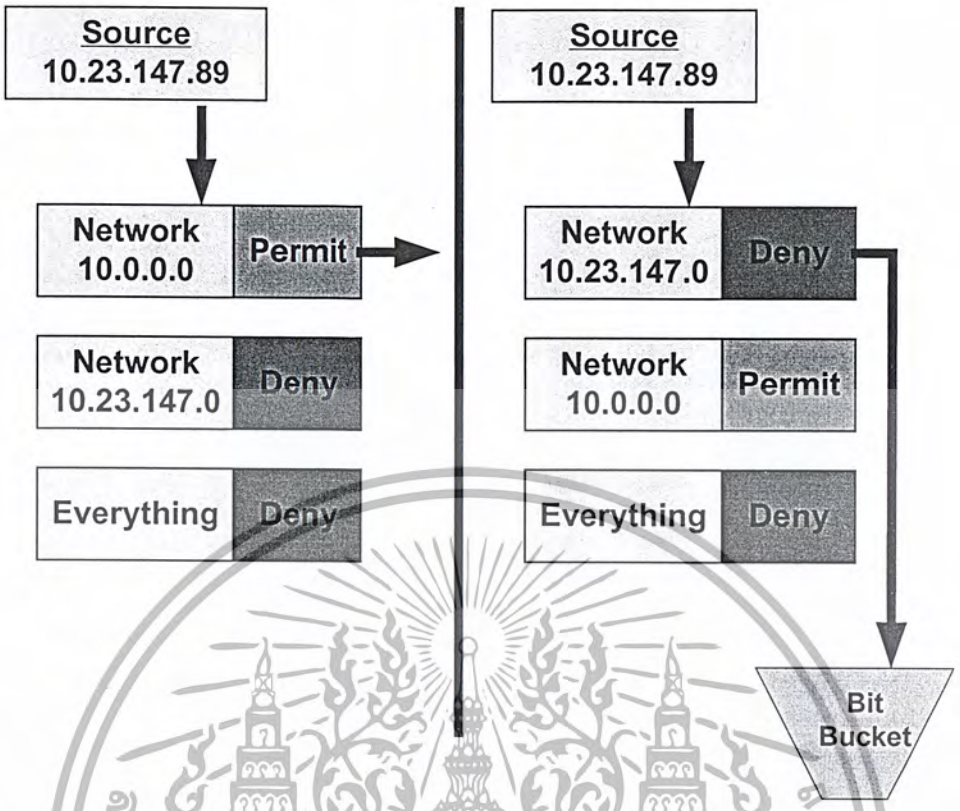
9.2 รูปแบบของ แอคเซสลิสต์

ตัวอย่างการคอนฟิกค่าของแอคเซสลิสต์

access-list 9 deny 10.23.147.0 0.0.0.255

access-list 9 permit 10.0.0.0 0.255.255.255

จากการคอนฟิกค่าของแอคเซสลิสต์ต่อแสดงได้ในรูปที่ 7-4



รูปที่ 9-4 ตัวอย่างการคอนฟิกลำดับขั้นของแอคเซสลิสต์ที่ถูกต้องและไม่ถูกต้อง

ในทุกๆชั้นของแอคเซสลิสต์จะแทนด้วยหนึ่งคำสั่งการคอนฟิกแต่จะเห็นว่ามีเลข 9 ในทั้งสองบรรทัดในการคอนฟิกตามตัวอย่าง ตัวเลขนี้เรียกว่า หมายเลขของแอคเซสลิสต์ ซึ่งจุดประสงค์เพื่อ

- เพื่อเชื่อมต่อหลายคำสั่งให้อยู่ในแอคเซสลิสต์เดียวกัน และเพื่อความแตกต่างระหว่างแอ็กเซสลิสต์ชุดอื่นที่อาจมีการคอนฟิกไว้ก่อน
- ทำให้เราเตอร์สามารถแยกความแตกต่างระหว่างรูปแบบของแอคเซสลิสต์แบบต่างๆได้ อย่างเช่นในซิสโก้ ไอโอเอส (IOS) มีแอคเซสลิสต์ จะมี IP , IPX , AppleTalk รวมทั้งโพรโตคอลต่างๆ มากมาย

Access List Type	Range
Standard IP	1-99
Extended IP	100 – 199
Ethernet type code	200-299
Ethernet address	700-799
Transparent bridging (protocol type)	200-299
Transparent bridging (vendor code)	700-799
Extended Transparent bridging	1100-1199
DECnet and extended DECnet	300-399
XNS	400-499
Extended XNS	500-599
AppleTalk	600-699
Source-route bridging (protocol type)	200-299
Source-route bridging (vendor code)	700-799
Standard IPX	800-899
Extended IPX	900-999
IPX SAP	1000-1099
NLSP route summery	1200-1299
Standard VINES	1-99
Extended VINES	100-199
Simple VINES	200-299

ตารางที่ 9-1 รูปแบบตัวเลขแอคเซสลิสต์ของซิสโก้

9.3 การแก้ไขค่าแอคเซสลิสต์

การใช้งานของแอคเซสลิสต์จากค่าเก่าที่ได้ทำการคอนฟิกไว้ก่อนแล้วสามารถทำได้โดยใช้คำสั่งการลบแอคเซสลิสต์ดังนี้

no access-list 101 permit tcp 10.2.5.4 0.0.0.255

ในที่นี้แอคเซสลิสต์หมายเลขที่ 101 จะถูกลบออกไป

อีกวิธีที่เราจะทำการคอนฟิกค่าแอคเซสลิสต์ได้คือ ทำการอัพโหลดไฟล์คอนฟิกที่เราได้เขียนเอาไว้ก่อน โดยใช้ ทีเอ็ฟทีพี เซิร์ฟเวอร์ (TFTP server) โดยรูปแบบในไฟล์ที่เราทำการเขียนไว้ควรจะมี ประโยคที่ทำการลบแอคเซสลิสต์ตัวเดิมออกก่อน ซึ่งมีรูปแบบดังนี้

no access-list #

เมื่อเครื่องหมาย # แทนหมายเลขของแอคเซสลิสต์ที่ต้องการใช้งาน ตัวอย่างมีดังนี้

no access-list 5
access-list 5 permit 10.0.0.1 0.0.0.0
access-list 5 permit 10.0.1.0 0.0.0.255
access-list deny any

ในบรรทัดแรก *no access-list 5* เป็นการลบแอคเซสลิสต์หมายเลข 5 ของเก่าลงเพื่อจะนำค่าการคอนฟิกใหม่ลงแทน เพื่อหลีกเลี่ยงปัญหาที่นำแอคเซสลิสต์ใหม่ไปต่อกับ แอคเซสลิสต์ที่มีอยู่เดิม

9.4 สแตนดาร์ด ไอพี แอคเซสลิสต์ (Standard IP Access Lists)

รูปแบบของการคอนฟิกแบบนี้คือ

access-list access-list-number {deny | permit} source [source-wildcard]

รูปแบบคำสั่งนี้จะใช้กับหมายเลขของแอคเซสลิสต์ตามตาราง คือ 1 ถึง 99 ส่วน *deny* กับ *permit* เป็นการเลือกการปฏิบัติงานเมื่อการเปรียบเทียบถูกต้อง *source* เป็น ไอพีแอดเดรสของต้นทาง และ ส่วน *source-wildcard* เป็นการกำหนดช่วงของ *source* นั้นเอง

ตัวอย่าง

access-list 1 permit 161.246.5.27 0.0.0.0
access-list 1 permit 161.246.5.15 0.0.0.0
access-list 1 deny 161.246.5.0 0.0.0.255
access-list 1 permit 161.246.0.0 0.0.31.255
access-list 1 deny 161.246.0.0 0.0.255.255
access-list permit 0.0.0.0 255.255.255.255

จากตัวอย่างสองบรรทัดแรกเป็นการกำหนด ให้หมายเลขโฮสต์ที่กำหนดไว้คือ 161.246.5.27 และ 161.246.5.15 สามารถผ่านได้โดยค่า 0.0.0.0 เป็นค่าไวร์คาร์ด *(wildcard)* ซึ่งจะเรียกอีกอย่างว่า อินเวิร์สแมส *(inverse mask)* ส่วนบรรทัดที่สามเป็นการป้องกันโฮสต์อื่นที่มาจากสับเน็ต 161.246.5.0 บรรทัดที่สี่ บอกว่าให้หมายเลขโฮสต์ตั้งแต่ 161.246.0.1 ถึง 161.246.31.255 ผ่านไปได้ โดนตัว อินเวิร์สแมส เป็นตัวกำหนดช่วงของแอดเดรส บรรทัดที่ห้าเป็นการป้องกันโฮสต์อื่นๆจากสับเน็ต 161.246.0.0 และบรรทัดสุดท้ายเป็นการอนุญาตให้โฮสต์อื่นๆผ่านไปได้

9.5 เอกซ์เทนเด็ด ไอพี แอคเซสลิสต์(Extended IP Access Lists)

รูปแบบของการคอนฟิกแบบนี้คือ

access-list access-list-number { deny | permit } protocol source source-wildcard destination destination-wildcard

รูปแบบคำสั่งนี้จะใช้กับหมายเลขของแอคเซสลิสต์ตามตาราง คือ 100 ถึง 199 ส่วนที่เพิ่มเข้ามาเป็นโพรโตคอล ซึ่งสามารถกำหนดได้ดังนี้ *eigrp ,gre , icmp , igmp , igrp , ip , ipinip , nos ,ospf , tcp , udp* โดยถ้าเรากำหนดค่าโพรโตคอล นี้เป็นไอพี *(ip)* จะสามารถใช้งานได้กับทุกๆโพรโตคอล ที่กล่าวมา ส่วนค่า *destination* และ *destination-wildcard* เป็นการกำหนดค่าปลายทางและช่วงของปลายทางนั้นเอง

ตัวอย่าง

access-list 101 permit ip 172.22.30.6 0.0.0.0 10.0.0.0 0.255.255.255
access-list 101 permit ip 172.22.30.95 0.0.0.0 10.11.12.0 0.0.0.255
access-list 101 deny ip 172.22.30.0 0.0.0.255 192.168.18.27 0.0.0.0
access-list 101 permit ip 172.22.0.0 0.0.31.255 192.168.18.0 0.0.0.255
access-list 101 deny ip 172.22.0.0 0.0.255.255 192.168.18.64 0.0.0.63
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255

บรรทัดที่ 1 กำหนดว่าเป็นไอพีแพ็กเก็ต *(IP Packet)* ที่มีหมายเลขต้นทางเป็น 172.22.30.6 และหมายเลขปลายทางเป็นเครือข่าย 10.0.0.0 ได้รับการอนุญาต *(permit)*

บรรทัดที่ 2 กำหนดว่าไอพีแพ็กเก็ต ที่หมายเลขต้นทางเป็น 172.22.30.95 และหมายเลขปลายทางมีเครือข่ายที่มีสับเน็ตเป็น 10.11.12.0/24 ได้รับการได้รับการอนุญาต

บรรทัดที่ 3 กำหนดว่าเป็นไอพีแพ็กเก็ต ที่มีหมายเลขต้นทางอยู่ในสับเน็ต 172.22.30.0/24 และ หมายเลขปลายทางมีแอดเดรสเป็น 192.168.18.27 จะถูกกำจัดไป

บรรทัดที่ 4 กำหนดว่าเป็นไอพีแพ็กเก็ตที่มีหมายเลขต้นทางระหว่าง 172.22.0.0 และ 172.22.31.255 และมีหมายเลขปลายทางอยู่ในเครือข่าย 192.168.18.0 จะได้รับการอนุญาต

บรรทัดที่ 5 กำหนดว่าเป็นไอพีแพ็กเก็ตที่มีหมายเลขต้นทางอยู่ในเครือข่าย 172.22.0.0 และมี หมายเลขปลายทางเป็น 26 บิตแรกของ 192.168.18.64 จะถูกกำจัดออกไป

บรรทัดที่ 6 กำหนดว่าเป็นไอพีแพ็กเก็ต ที่เหลือจะได้รับการอนุญาต

9.6 ทีซีพี แอคเซสลิสต์ (TCP Access List)

รูปแบบของการคอนฟิกแบบนี้คือ

access-list access-list-number { deny | permit } tcp source source-wildcard [operator [port]] destination destination-wildcard [operator [port]]

ในที่นี้จะเป็นเอ็กซ์เทนเด็ดแอคเซสลิสต์(*Extended access list*)ประเภทที่มีโพรโตคอลเป็นทีซีพีซึ่งสามารถทำการตรวจสอบ แพ็กเก็ตที่เป็นทีซีพีและสามารถตรวจสอบหมายเลขพอร์ต ได้

- Operator เป็นการกำหนดการเปรียบเทียบสำหรับหมายเลขพอร์ต ได้แก่ eq , neq , gt , lt ซึ่งความหมายคือ เท่ากับ ไม่เท่ากับ มากกว่า และ น้อยกว่าตามลำดับ
- Port เป็นหมายเลขพอร์ตที่กำหนด ยกตัวอย่างเช่น เทลเน็ต (23) , เอฟทีพี (20 และ 21) , เอสเอ็นทีพี (25) เป็นต้น

ตัวอย่าง

access-list 110 permit tcp 10.0.0.0 0.255.255.255 eq 80 172.22.144.0 0.0.0.255 eq 80

จากตัวอย่าง เป็นการอนุญาต ซึ่งกำหนดโพรโตคอลเป็นทีซีพี โดยมีแอดเดรสต้นทางเป็น 10.0.0.0 ที่เป็นการติดต่อผ่านพอร์ต 80 ไปยังปลายทาง ที่แอดเดรสปลายทาง 172.22.144.0.0/24

9.7 ยูดีพี แอคเซสลิสต์ (UDP Access List)

รูปแบบของการคอนฟิกแบบนี้คือ

access-list access-list-number { deny | permit } udp source source-wildcard [operator [port]] destination destination-wildcard [operator [port]]

ในที่นี้จะเป็น เอ็กซ์เทนเด็ดแอคเซสลิสต์ *(Extended access list)* ประเภทที่มีโพรโตคอล เป็น ยูดีพี โดยโพรโตคอลนี้จะมีลักษะการเรียกใช้งานเหมือนกับโพรโตคอลทีซีพีแต่ลักษณะที่ต่างกันคือโพรโตคอล ยูดีพี จะใช้การทำงานแบบคอนเน็กชั่นเลส *(Connection less)* ในการติดต่อระหว่างเครื่อง ซึ่งหมายความว่าไม่มีการสร้างเส้นทางการเชื่อมต่อไว้ก่อน

ตัวอย่าง

access-list 109 permit udp 10.0.0.0 0.0.0.255 11.0.0.0 0.0.0.255 eq 161

จากตัวอย่าง เป็นการอนุญาต ซึ่งกำหนดโพรโตคอลเป็นยูดีพี โดยมีแอดเดรสต้นทางเป็น 10.0.0.0 ที่เป็นการติดต่อผ่านไปยังปลายทาง ที่แอดเดรสปลายทาง 172.22.144.0.0/24 ผ่านพอร์ต 161 หรือเป็นเอสเอ็นทีพี แพ็คเก็ต

9.8 ไอซีเอ็มพี แอคเซสลิสต์ (ICMP Access List)

รูปแบบของการคอนฟิกแบบนี้คือ

access-list access-list-number { deny | permit } icmp source source-wildcard destination destination-wildcard [icmp-type [icmp-code]]

ในที่นี้จะเป็นเอ็กซ์เทนเด็ดแอคเซสลิสต์ *(Extended access list)* ประเภทที่มีโพรโตคอล เป็นไอซีเอ็มพี โดยโพรโตคอลนี้จะไม่มีการกำหนดพอร์ตในการเชื่อมต่อของแอดเดรสต้นทางและปลายทางโดย โพรโตคอลแบบไอซีเอ็มพี จะเป็นโพรโตคอลที่อยู่ในชั้นเครือข่าย *(network layer)* ซึ่งสามารถทำการตรวจสอบข้อความของไอซีเอ็มพีซึ่งประกอบด้วย

- icmp-type เป็นตัวเลขระหว่าง 0 – 255 โดยสามารถดูรายละเอียดได้ใน อาร์เอ็ฟซี(RFC) 1700
- icmp-code เป็นการกำหนดสับเซ็ต ของรูปแบบไอซีเอ็มพีแพ็กเก็ต ซึ่งมีค่าระหว่าง 0 – 255

ตัวอย่าง

access-list 111 deny icmp 172.22.0.0 0.0.255.255 0.0.0.0 255.255.255.255 3 9

จากตัวอย่างเป็นการป้องกันแพ็กเก็ตไอซีเอ็มพีที่มีเครือข่ายแอดเดรสต้นทาง เป็น 172.22.0.0 ไปยังปลายทางทุกตัวด้วยซึ่งเป็น ICMP destination unreachable packet (type 3) และมีโค้ดเป็นเลข 9 ซึ่งมีความหมายว่า Network Administratively Prohibited

9.8 ไอซีเอ็มพี แอคเซสลิสต์ (ICMP Access List)

ในการกำหนดค่าแอคเซ็สลิสต์ต้องกำหนดให้แต่ละอินเทอร์เฟสด้วยคำสั่ง

ip access-group access-list-number { in | out }

ซึ่งคำสั่งนี้เป็นการกำหนดค่าความปลอดภัยหรือตรวจสอบให้กับอินเทอร์เฟสที่เรียกใช้คำสั่งนี้โดยการกำหนดหมายเลขของแอคเซสลิสต์ จะเป็นการเลือกหมายเลขของแอคเซสลิสต์ที่ทำการคอนฟิกกูเลชั่นไว้ โดยแต่ละอินเทอร์เฟสจะกำหนดการตรวจสอบว่าจะตรวจสอบในขณะที่แพ็กเก็ตเข้ามาหรือออกไปโดยการกำหนด *in* หรือ *out*

ตัวอย่าง การใช้งาน

Router(config-int)# ip access-group 15 in

หมายถึงการกำหนดแอคเซสลิสต์กลุ่มที่มีหมายเลขเป็น 15 ให้กับอินเทอร์เฟสโดยจะตรวจสอบทางเข้าของอินเทอร์เฟส

บทที่ 10
การออกแบบและการสร้างโปรแกรม

10.1 โครงสร้างของโปรแกรม (Design)

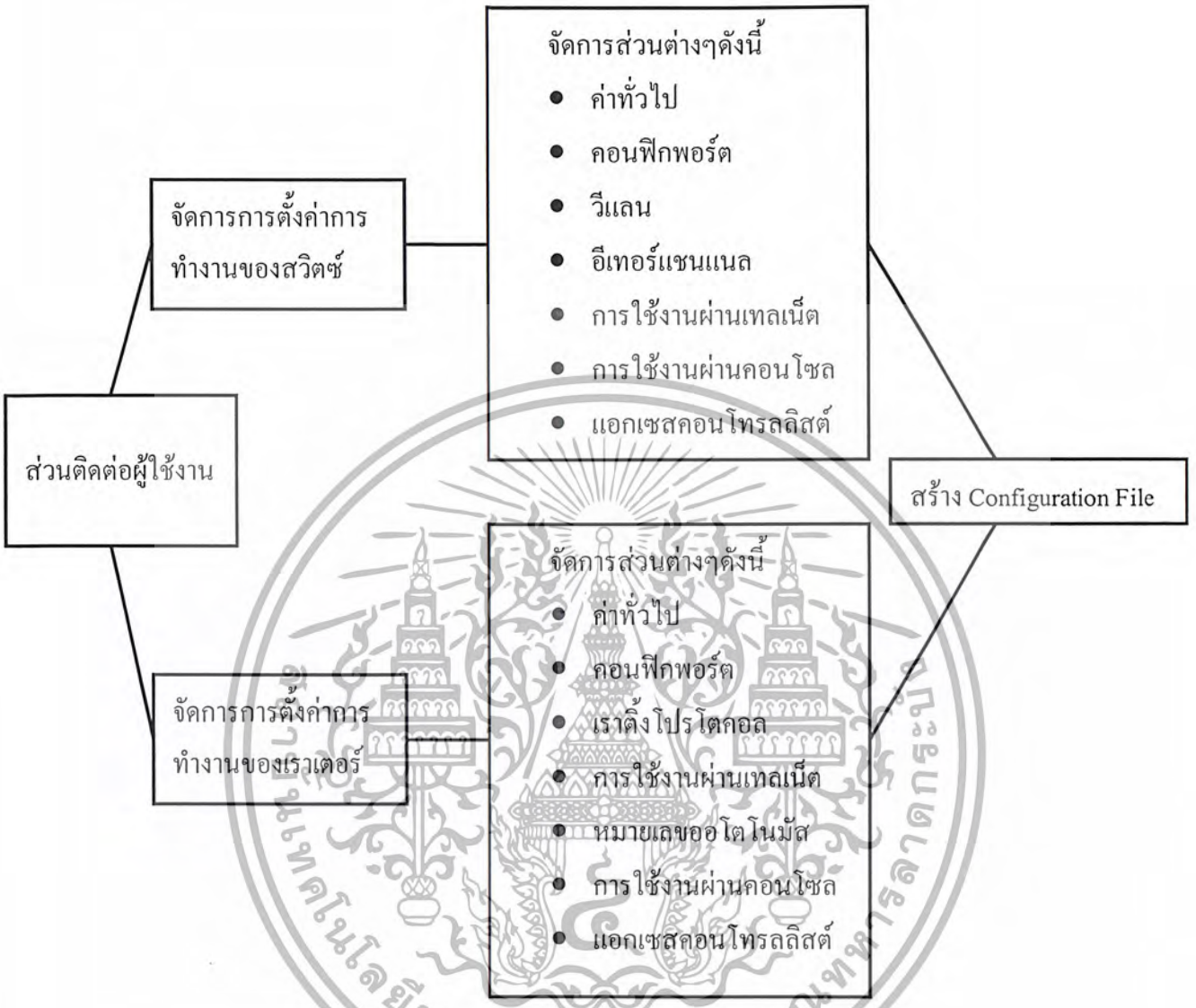
โปรแกรมนี้ ออกแบบโดยใช้หลักการของออบเจ็กต์โอเรียนเต็ดโปรแกรมมิ่ง (Object Oriented Programming) เพื่อให้โปรแกรมง่ายต่อการเพิ่มเติมและดัดแปลงโปรแกรม โดยแบ่งการทำงานของโปรแกรมออกเป็นส่วนย่อย ๆ เป็นคลาสต่าง ๆ แต่ละคลาสจะมีหน้าที่การทำงานที่แตกต่างกันดังนี้

- Frame1 เป็นคลาสที่สร้างหน้าจออินเทอร์เฟส คอยติดต่อกับผู้ใช้งาน โดยหน้าอินเทอร์เฟสจะประกอบไปด้วยเมนูบาร์ ทูลบาร์ และส่วนที่ใช้แสดงภาพเครือข่ายจำลองในรูปแบบกราฟฟิก
- Switch เป็นคลาสที่ประกอบไปด้วยข้อมูลต่างๆ ของสวิตช์เช่น ชื่อสวิตช์ อินเทอร์เฟสของสวิตช์ โดยฟังก์ชันการทำงานของคลาสนี้จะมีฟังก์ชันสำหรับการสร้างConfiguration Fileของสวิตช์ด้วย
- Router เป็นคลาสที่ประกอบไปด้วยข้อมูลต่างๆ ของเราเตอร์เช่น ชื่อเราเตอร์ อินเทอร์เฟสของเราเตอร์โดยฟังก์ชันการทำงานของคลาสนี้จะมีฟังก์ชันสำหรับการสร้างConfiguration Fileของเราเตอร์ด้วย
- SwitchImg เป็นคลาสที่เก็บข้อมูลเกี่ยวกับภาพของสวิตช์แต่ละตัว
- RouterImg เป็นคลาสที่เก็บข้อมูลเกี่ยวกับภาพของเราเตอร์แต่ละตัว
- Connection เป็นคลาสที่เก็บข้อมูลการเชื่อมต่อของอุปกรณ์แต่ละตัว
- DesignAreaPanel เป็นคลาสที่แสดงเครือข่ายที่เราสร้างขึ้น แสดงตำแหน่งและการเชื่อมต่อของอุปกรณ์แต่ละตัว และใช้ในการเรียกเมนูป๊อปอัพของแต่ละอุปกรณ์
- IdealSwitch เป็นคลาสที่ใช้เก็บข้อมูลที่จำเป็นในการเพิ่มสวิตช์ใหม่ที่ต้องการ
- IdealRouter เป็นคลาสที่ใช้เก็บข้อมูลที่จำเป็นในการเพิ่มเราเตอร์ใหม่ที่ต้องการ
- ส่วนคลาสอื่นๆจะมีสองประเภทหลักๆคือ คลาสที่ใช้ในการเก็บข้อมูลต่างๆเพื่อเรียกมาใช้ และคลาสที่เป็นการสร้างไดอะล็อกขึ้นมาเพื่อติดต่อกับผู้ใช้งานสามารถใช้งานได้ง่ายขึ้น

โดยคลาสที่สร้างขึ้น สามารถแบ่งได้เป็น 3 ส่วนหลักๆคือ

1. ส่วนติดต่อกับผู้ใช้งาน
2. ส่วนการตั้งค่าการทำงานของสวิตช์
3. ส่วนการตั้งค่าการทำงานของเราเตอร์

โดยคลาสต่างๆมีการจัดการดังนี้



รูปที่ 10-1 การออกแบบโปรแกรม

10.2 ส่วนติดต่อกับผู้ใช้งาน

แบ่งได้เป็นหลายส่วนคือ

1. ส่วนหน้าจอหลักของโปรแกรม ประกอบด้วยทูลบาร์ที่รวมเอาการทำงานต่างๆ ได้แก่ การเพิ่มสวิตซ์ การเพิ่มเราเตอร์ การสร้างสวิตซ์และเราเตอร์ขึ้นมาใหม่ การสร้างการเชื่อมต่อ การยกเลิกการเชื่อมต่อ และการเรียกดูตัวอย่างการใช้งาน สำหรับอีกส่วนคือส่วนของหน้าจอแสดงแผนภาพเครือข่าย ซึ่งจะแสดงว่ามีอุปกรณ์ใดบ้าง และมีการติดต่อกันอย่างไร โดยส่วนนี้จะใช้ในการเรียกหน้าจอต่างๆที่ใช้ในการปรับแต่งค่าต่างๆให้กับอุปกรณ์นั้นๆด้วย โดยในส่วนนี้จะมี method การทำงานในคลาส Frame1 และ คลาส DesignAreaPanel โดย method การทำงานที่สำคัญๆของคลาสทั้งสองคลาสมีดังนี้

<u>คลาส Frame1</u>

- Frame1() เป็นคอนสตรัคเตอร์ของคลาส ใน method นี้จะแสดงหน้าจอโปรแกรมหลักทั้งส่วนของเมนูบาร์ และทูลบาร์
- newSwitch() เป็น method สำหรับสร้างสวิตช์แต่ละตัวขึ้นมาใหม่ โดยจะมีการตรวจสอบชนิดของสวิตช์ว่าต้องสร้างสวิตช์ชนิดไหนขึ้นมาใหม่เพื่อเพิ่มเข้าไปในเวกเตอร์ vSwitch ซึ่งเป็นเวกเตอร์สำหรับเก็บสวิตช์ทั้งหมดที่สร้างขึ้น
- newRouter() เป็น method สำหรับสร้างเราเตอร์แต่ละตัวขึ้นมาใหม่ โดยจะมีการตรวจสอบชนิดของเราเตอร์ว่าต้องสร้างเราเตอร์ชนิดไหนขึ้นมาใหม่เพื่อเพิ่มเข้าไปในเวกเตอร์ vRouter ซึ่งเป็นเวกเตอร์สำหรับเก็บเราเตอร์ทั้งหมดที่สร้างขึ้น
- seveFile() เป็น method ที่ใช้ในการเก็บไฟล์ข้อมูลของสวิตช์และเราเตอร์ชนิดใหม่ที่เพิ่มเข้าไป โดยจะเรียกทำงานตอนปิดโปรแกรมเพื่อเก็บสวิตช์ที่เพิ่มเข้าไปไว้ใช้
- ส่วน method อื่นๆ เป็น method สำหรับจัดการการกระทำของผู้ใช้สำหรับแต่ละคอมโพเน็นท์ ซึ่งเป็นการจัดการการกระทำที่ผู้ใช้กระทำกับคอมโพเน็นท์หนึ่งๆ ในขณะที่คอมโพเน็นท์หนึ่งๆ ถูกโฟกัสอยู่ เช่น jNewButton1_actionPerformed() เป็นการจัดการกับการกระทำของผู้ใช้กับปุ่มเพิ่มสวิตช์

คลาส DesignAreaPanel

- DesignAreaPanel() เป็นคอนสตรัคเตอร์ของคลาส
- paintComponent() เป็นคลาสที่ใช้ในการวาดรูปอุปกรณ์และการเชื่อมต่อต่างๆ
- drawLine() ใช้วาดการเชื่อมต่อของอุปกรณ์ต่างๆ
- this_mousePressed() เป็น method ที่ใช้ในการจัดการกับการคลิ้กเมาส์ เพื่อรับรู้ว่าเราคลิ้กเมาส์ที่อุปกรณ์ตัวไหน
- this_mouseReleased() เป็น method ที่ใช้ในการจัดการกับการปล่อยเมาส์
- this_mouseDragged() เป็น method ที่ใช้ในการจัดการกับการคลิ้กเมาส์ และเลื่อนเมาส์ไปพร้อมกัน เพื่อใช้ในการเคลื่อนย้ายอุปกรณ์
- ส่วน method อื่นๆ เป็น method สำหรับจัดการการกระทำของผู้ใช้สำหรับแต่ละคอมโพเน็นท์ ซึ่งเป็นการจัดการการกระทำที่ผู้ใช้กระทำกับคอมโพเน็นท์หนึ่งๆ ในขณะที่คอมโพเน็นท์หนึ่งๆ ถูกโฟกัสอยู่ เช่น configSW_actionPerformed() เป็นการจัดการกับการกระทำของผู้ใช้กับปุ่มคอนฟิกสวิตช์

2. หน้าจอการสร้างการเชื่อมต่อของอุปกรณ์ จะเป็นส่วนที่ใช้เลือกว่าจะสร้างการเชื่อมต่อกันระหว่างอุปกรณ์ตัวไหนที่พอร์ตไหน โดยจะเก็บการเชื่อมต่อที่สร้างขึ้นไว้ในคลาส Frame1 โดยในส่วนนี้ จะมี method การทำงานจะอยู่ในคลาส ConnectionDialog โดยมี method การทำงานดังนี้
   - ConnectionDialog() เป็นคอนสตรัคเตอร์ของคลาสและสร้างไดอะล็อกขึ้นมา
   - swNameBox1_actionPerformed() ใช้ในการจัดการตอนเลือกอุปกรณ์ว่าจะให้มีพอร์ตไหนของอุปกรณ์ตัวนั้นบ้างที่สามารถเลือกได้

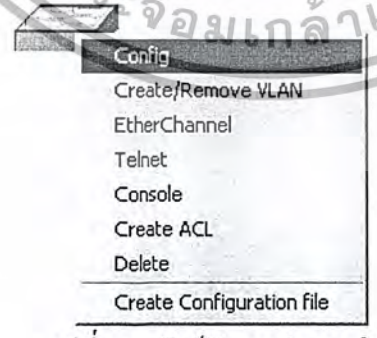
- swNameBox2_actionPerformed()ใช้ในการจัดการตอนเลือกอุปกรณ์ว่าจะให้มีพอร์ตไหนของอุปกรณ์ตัวนั้นบ้างที่สามารถเลือกได้
- showDialog() เป็น method ที่ใช้ในการแสดงหน้าจอ
- ส่วน method อื่นๆ เป็น method สำหรับจัดการการกระทำของผู้ใช้สำหรับแต่ละคอมโพเน็นท์ ซึ่งเป็นการจัดการการกระทำที่ผู้ใช้กระทำกับคอมโพเน็นท์หนึ่งๆ ในขณะที่คอมโพเน็นท์หนึ่งๆ ถูกโฟกัสอยู่

3. หน้าจอการยกเลิกการเชื่อมต่อ เป็นส่วนที่แสดงการเชื่อมต่อของอุปกรณ์ต่างๆ และสามารถเลือกได้ว่าจะทำการยกเลิกการเชื่อมใด โดยในการแสดงการเชื่อมต่อจะนำมาจากคลาส Frame1 และเมื่อทำการยกเลิกการเชื่อมต่อใดๆแล้วก็จะทำการเปลี่ยนแปลงที่คลาส Frame1 โดยการทำงานในส่วนนี้จะอยู่ในคลาส DisconnectionDialog โดยมี method การทำงานดังนี้
   - DisconnectionDialog() เป็นคอนสตรัคเตอร์ของคลาสและสร้างไดอะล็อกขึ้นมา
   - setList() เป็น method ที่ใช้ในการอัพเดตเวกเตอร์ vConnection ในคลาส Frame1
   - cancel_actionPerformed() เป็น method ที่ใช้ในการยกเลิกการใช้งานไดอะล็อกนี้
   - disConnect_actionPerformed() เป็น method ที่ใช้ในการยกเลิกการเชื่อมต่อที่ถูกเลือก

4. ส่วนติดต่อกับผู้ใช้งานส่วนอื่นๆนั้นจะเป็นส่วนที่ใช้ในการปรับแต่งค่าต่างๆของอุปกรณ์ สวิตซ์ และเราเตอร์โดยจะขออธิบายการทำงานในหัวข้อต่อๆไป

10.3 ส่วนการจัดการสวิตซ์

ในส่วนนี้จะเป็นการกระทำต่างๆที่ทำกับสวิตซ์ได้แก่ การคอนฟิกสวิตซ์ คอนฟิกพอร์ต การสร้างแลนเสมือน การสร้างพอร์ตแชนแนล การกำหนดรหัสผ่านการเทลเน็ต การกำหนดรหัสผ่านไลน์คอนโซล การสร้างแอกเซสคอนโทรลลิสต์ การสร้างConfiguration File การลบสวิตซ์ โดยการกระทำต่างๆจะมีการเรียกใช้คลาสดังนี้



รูปที่ 10-2 ฟังก์ชันการทำงานต่างๆของสวิตซ์

10.3.1 การคอนฟิกสวิตซ์

เป็นส่วนที่ใช้กำหนดชื่อให้อุปกรณ์ กำหนดไอโอเอสเวอร์ชัน กำหนดดีฟอล์ทเกตเวย์ กำหนดพาสเวิร์ด และการคอนฟิกพอร์ตต่างๆ โดยส่วนนี้อยู่ในคลาส SwitchDialog ซึ่งมี method การทำงานดังนี้

- SwitchDialog() เป็นคอนสตรัคเตอร์ของคลาสสำหรับสร้างค่าเริ่มต้นต่างๆของไดอะล็อก
- chkSW() ใช้ตรวจสอบว่าเป็นสวิตซ์ตัวไหนที่เราเลือกเพื่อที่จะเซตค่าต่างๆให้ถูกต้อง
- showDialog1() ใช้ในการแสดงไดอะล็อกนี้โดยจะเซตขนาดตามที่กำหนดไว้
- chkIP() ใช้ตรวจสอบความถูกต้องของไอพีแอดเดรส
- correctIP() ใช้แก้ไขไอพีแอดเดรสให้ถูกต้อง
- ok_actionPerformed() เป็น method ที่ใช้ยืนยันการเปลี่ยนแปลงค่าต่างๆที่เรากำหนดใน GUI
- cancel_actionPerformed() เป็น method ที่ใช้ยกเลิกการเปลี่ยนแปลงค่าต่างๆ
- proprety_actionPerformed() เป็น method ที่ใช้ในการเรียกหน้าจอปรับแต่งค่าให้พอร์ตต่างๆ

โดยการคอนฟิกพอร์ตจะมีสามารถกำหนดได้ว่าจะให้เป็นโหมดแอกเซส หรือโหมดทรั้ง โดยถ้าเลือกเป็นโหมดแอกเซสจะสามารถกำหนดคอสและไพรออริตีได้ ถ้าเป็นโหมดทรั้งจะสามารถเลือกแลนเสมือนได้นอกจากนี้ยังสามารถกำหนดไอพีแอดเดรส ซับเนตมาร์คและอีเทอร์แชนแนล method การทำงานในส่วนนี้จะอยู่ในคลาส ConfigPort โดยมี method การทำงานดังนี้

- ConfigPort() เป็นคอนสตรัคเตอร์ของคลาสสำหรับสร้างค่าเริ่มต้นต่างๆของไดอะล็อก
- config() ใช้ตรวจสอบว่าพอร์ตที่จะคอนฟิกเป็นพอร์ตไหนของสวิตซ์ตัวไหน
- showDialog()ใช้ในการแสดงไดอะล็อกนี้โดยจะเซตขนาดตามที่กำหนดไว้
- chkInteger() ใช้ตรวจสอบว่าสตริงที่รับเข้ามาเป็นตัวเลขหรือไม่
- chkIP() ใช้ตรวจสอบความถูกต้องของไอพีแอดเดรส
- chkSubnet() ใช้ตรวจสอบความถูกต้องของซับเนตมาร์ค
- correctIP() ใช้แก้ไขไอพีแอดเดรสให้ถูกต้อง
- isInThisVector() ตรวจสอบว่าออบเจคที่ส่งมาอยู่ในเวกเตอร์ที่ส่งมาหรือไม่
- rankInThisVector() ตรวจสอบตำแหน่งของออบเจคในเวกเตอร์ที่ส่งมา
- ok_actionPerformed() เป็น method ที่ใช้ในการเปลี่ยนแปลงค่าตามที่ทำการเซตในหน้าจอ GUI
- cancel_actionPerformed()เป็น method ที่ใช้ยกเลิกการเปลี่ยนแปลงค่าต่างๆ
- ส่วน method อื่นๆจะใช้สำหรับการทำงานต่างๆบนหน้าจอ GUI เช่นการเซตโหมด การเลือกแลนเสมือน เป็นต้น

10.3.2 การสร้างแลนเสมือน

เป็นส่วนที่ใช้ในการสร้างและลบแลนเสมือน การแก้ไขรายละเอียดต่างๆของแลนเสมือน เช่น การใส่ไอพีแอดเดรส การกำหนดแอ็กเซสคอนโทรลลิสต์ method การทำงานในส่วนนี้จะอยู่ในคลาส CreateVlan โดยมี method การทำงานดังนี้

- CreateVlan() เป็นคอนสตรัคเตอร์ของคลาสสำหรับสร้างค่าเริ่มต้นต่างๆของไดอะล็อก
- chkSW()ใช้ตรวจสอบว่าเป็นสวิตซ์ตัวไหนที่เราเลือกเพื่อที่จะเซตค่าต่างๆให้ถูกต้อง
- showDialog1() ใช้ในการแสดงไดอะล็อกนี้โดยจะเซตขนาดตามที่กำหนดไว้
- create_actionPerformed() เป็น method ที่ใช้ในการสร้างแลนเสมือนโดยจะสร้างตามที่เราใส่ค่าไว้

- ส่วน method อื่นๆจะเป็น method ที่ใช้ในการลบ แลนเสมือน การตกลงในการสร้าง แลนเสมือน และการแก้ไขรายละเอียดของแลนเสมือน

สำหรับการแก้ไขรายละเอียดของแลนเสมือนได้ผ่านทางหน้าจอแก้ไขแลนเสมือน โดยการทำงานในการแก้ไขแลนเสมือนจะอยู่ในคลาส EditVlan โดยมี method การทำงานดังนี้

- EditVlan() เป็นคอนสตรัคเตอร์ของคลาสสำหรับสร้างค่าเริ่มต้นต่างๆของไดอะล็อก
- setVlan() ใช้บอกให้รู้ว่าเป็นแลนเสมือนอะไรของสวิตช์ตัวไหน
- showDialog()ใช้ในการแสดงไดอะล็อกนี้โดยจะเซตขนาดตามที่กำหนดไว้
- chkInteger() ใช้ตรวจสอบว่าสตริงที่รับเข้ามาเป็นตัวเลขหรือไม่
- chkIP() ใช้ตรวจสอบความถูกต้องของไอพีแอดเดรส
- chkSubnet() ใช้ตรวจสอบความถูกต้องของซับเนตมาร์ค
- correctIP() ใช้แก้ไขไอพีแอดเดรสให้ถูกต้อง
- ok_actionPerformed() ใช้ตกลงการเปลี่ยนแปลงค่าที่ได้กระทำ
- cancel_actionPerformed() ใช้ยกเลิกการเปลี่ยนแปลงทั้งหมด
- aclButton_actionPerformed() ใช้เรียกหน้าจอเอซีแอลเพื่อใช้ในการเซตเอซีแอลให้แลนเสมือน

10.3.3 การกำหนดอีเทอร์แชนแนล

เป็นการกำหนดให้สามารถใช้อีเทอร์แชนแนลที่กำหนดได้โดยจะกำหนดให้มีให้เลือกใช้งานเพียง 8 กลุ่มโดยสามารถเลือกว่าจะใช้งานหรือไม่ใช้งานกลุ่มไหนโดยการทำงานจะอยู่ในคลาส EtherChannel โดยมี method การทำงานดังนี้

- EtherChannel() เป็นคอนสตรัคเตอร์ของคลาสสำหรับสร้างค่าเริ่มต้นต่างๆของไดอะล็อก
- chkSW() ใช้ตรวจสอบว่าเป็นสวิตช์ตัวไหนที่เราเลือกเพื่อที่จะเซตค่าต่างๆให้ถูกต้อง
- showDialog() ใช้ในการแสดงไดอะล็อกนี้โดยจะเซตขนาดตามที่กำหนดไว้
- isInThisVector() ตรวจสอบว่าออบเจคที่ส่งมาอยู่ในเวกเตอร์ที่ส่งมาหรือไม่
- rankInThisVector() ตรวจสอบตำแหน่งของออบเจคในเวกเตอร์ที่ส่งมา
- ok_actionPerformed() เป็น method ที่ใช้ในการตกลงการเปลี่ยนแปลงค่าที่ได้กระทำ
- cancel_actionPerformed() เป็น method ที่ใช้ในการยกเลิกการเปลี่ยนแปลงทั้งหมด

10.3.4 การกำหนดการใช้งานผ่านเทลเน็ต

ใช้ในการกำหนดการใช้งานผ่านเทลเน็ต การตั้งค่ารหัสผ่าน รวมถึงการกำหนดแอกเซสคอนโทรลลิสต์โดยการทำงานจะอยู่ในคลาส TelnetSwitch โดยมี method การทำงานดังนี้

- TelnetSwitch() เป็นคอนสตรัคเตอร์ของคลาสสำหรับสร้างค่าเริ่มต้นต่างๆของไดอะล็อก
- setSwitch() ใช้บอกว่าเป็นสวิตช์ตัวไหน
- showDialog()ใช้ในการแสดงไดอะล็อกนี้โดยจะเซตขนาดตามที่กำหนดไว้
- ok_actionPerformed() เป็น method ที่ใช้ในการตกลงการเปลี่ยนแปลงค่าที่ได้กระทำ

- cancel_actionPerformed() เป็น method ที่ใช้ในการยกเลิกการเปลี่ยนแปลงทั้งหมด

10.3.5 การกำหนดการทำงานผ่านทางคอนโซล

ใช้ในการกำหนดการใช้งานผ่านางคอนโซล การตั้งค่ารหัสผ่าน รวมถึงการกำหนดแอกเซสคอนโทรลลิสต์ โดยการทำงานจะอยู่ในคลาส ConsoleSwitch โดยมี method การทำงานดังนี้

- ConsoleSwitch() เป็นคอนสตรัคเตอร์ของคลาสสำหรับสร้างค่าเริ่มต้นต่างๆของไดอะล็อก
- setSwitch() ใช้บอกว่าเป็นสวิตซ์ตัวไหน
- showDialog()ใช้ในการแสดงไดอะล็อกนี้โดยจะเซตขนาดตามที่กำหนดไว้
- ok_actionPerformed() เป็น method ที่ใช้ในการตกลงการเปลี่ยนแปลงค่าที่ได้กระทำ
- cancel_actionPerformed() เป็น method ที่ใช้ในการยกเลิกการเปลี่ยนแปลงทั้งหมด

10.3.6 การสร้างแอกเซสคอนโทรลลิสต์

เป็นการสร้างแอกเซสคอนโทรลลิสต์ผ่านทางหน้าจอโดยสามารถ แก้ไข เพิ่มเติมได้ผ่านทางหน้าจอนี้โดยสามารถสร้างได้ทั้งแบบ Standard ACL และ Extend ACL ซึ่งถ้าหมายเลขเอซีแอลอยู่ระหว่าง 1-99 จะเป็น Standard ACL และถ้าหมายเลขเอซีแอลอยู่ระหว่าง 100 – 199 จะเป็น Extend ACLโดยการทำงานจะอยู่ในคลาส ACLDialog โดยมี method การทำงานดังนี้

- ACLDialog() เป็นคอนสตรัคเตอร์ของคลาสสำหรับสร้างค่าเริ่มต้นต่างๆของไดอะล็อก
- setRouter() ใช้บอกว่าเป็นเราเตอร์ตัวไหน
- setSwitch() ใช้บอกว่าเป็นสวิตซ์ตัวไหน
- showDialog() ใช้ในการแสดงไดอะล็อกนี้โดยจะเซตขนาดตามที่กำหนดไว้
- chkInteger()ใช้ตรวจสอบว่าสตริงที่รับเข้ามาเป็นตัวเลขหรือไม่
- create_actionPerformed() เป็น method ที่ใช้ในการสร้างเอซีแอล
- edit_actionPerformed() เป็น method ที่ใช้ในการแก้ไข ปรับแต่งค่าของเอซีแอล
- remove_actionPerformed() เป็น method ที่ใช้ในการลบเอซีแอลที่ไม่ต้องการ
- ok_actionPerformed() เป็น method ที่ใช้ในการตกลงการเปลี่ยนแปลงค่าที่ได้กระทำ
- cancel_actionPerformed()เป็น method ที่ใช้ในการยกเลิกการเปลี่ยนแปลงทั้งหมด

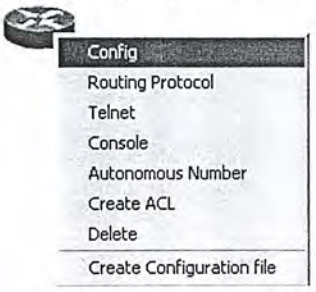
10.3.7 ในส่วนการสร้างConfiguration File

ในส่วนนี้เมื่อมีการเรียกใช้จะเรียก method genConfigFile()ในคลาสสวิตซ์มาใช้ โดย method นี้จะสร้างไฟล์ใหม่ขึ้นมาให้เราเซฟConfiguration Fileที่ได้ไปไว้ที่ไดเรกทอรี่ที่ต้องการ

10.4 ส่วนการจัดการเราเตอร์

ในส่วนนี้จะเป็นการกระทำต่างๆที่ทำกับเราเตอร์ได้แก่ การคอนฟิกเราเตอร์ คอนฟิกพอร์ต การกำหนดรหัสผ่านการในการใช้งานผ่านเทลเน็ต การกำหนดรหัสผ่านในการใช้งานผ่านคอนโซล การ

กำหนดเราติ้งโปรโตคอล การกำหนดหมายเลขออโตโนมัส การสร้างแอกเซสคอนโทรลลิสต์ การสร้าง Configuration File การลบเราเตอร์ โดยการกระทำต่างๆจะมีการเรียกใช้คลาสดังนี้



รูปที่ 10-3 หน้าจอสร้างแอกเซสคอนโทรลลิสต์

10.4.1 การคอนฟิกเราเตอร์

เป็นส่วนที่ใช้กำหนดชื่อให้อุปกรณ์ กำหนดไอโอเอสเวอร์ชัน กำหนดดีฟอล์ทเกตเวย์ กำหนดพาสเวิร์ด และการคอนฟิกพอร์ตต่างๆ โดยส่วนนี้อยู่ในคลาส RouterDialog ซึ่งมี method การทำงานดังนี้

- RouterDialog() เป็นคอนสตรัคเตอร์ของคลาสสำหรับสร้างค่าเริ่มต้นต่างๆของไดอะล็อก
- chkRT() ใช้ตรวจสอบว่าเป็นเราเตอร์ตัวไหนที่เราเลือกเพื่อที่จะเซตค่าต่างๆให้ถูกต้อง
- showDialog() ใช้ในการแสดงไดอะล็อกนี้โดยจะเซตขนาดตามที่กำหนดไว้
- chkIP() ใช้ตรวจสอบความถูกต้องของไอพีแอดเดรส
- correctIP() ใช้แก้ไขไอพีแอดเดรสให้ถูกต้อง
- setList() ใช้เซตค่าว่ามีพอร์ตอะไรบ้างที่สามารถคอนฟิกได้
- ok_actionPerformed() เป็น method ที่ใช้ยืนยันการเปลี่ยนแปลงค่าต่างๆที่เรากำหนดใน GUI
- cancel_actionPerformed() เป็น method ที่ใช้ยกเลิกการเปลี่ยนแปลงค่าต่างๆ
- proprety_actionPerformed() เป็น method ที่ใช้ในการเรียกหน้าจอปรับแต่งค่าให้พอร์ตต่างๆ

โดยการคอนฟิกพอร์ตของเราเตอร์จะมี 3 ประเภทคือ

1. พอร์ตอีเทอร์เนต
2. พอร์ตซับอินเทอร์เฟส
3. พอร์ตซีเรียล

โดยการคอนฟิกพอร์ตต่างๆจะทำงานดังนี้

1. พอร์ตอีเทอร์เนต เป็นการคอนฟิกพอร์ตอีเทอร์เนตโดยสามารถกำหนดค่าต่างๆดังนี้ ไอพีแอดเดรส สับเนต และการกำหนดแอกเซสคอนโทรลลิสต์ โดยการทำงานจะอยู่ในคลาส ConfigInfRouter โดยมีการทำงานดังนี้

- ConfigInfRouter() เป็นคอนสตรัคเตอร์ของคลาสสำหรับสร้างค่าเริ่มต้นต่างๆของไดอะล็อก
- setList() ใช้แสดงว่ามีแอกเซสคอนโทรลลิสต์ใดบ้างที่ถูกแอดสำหรับพอร์ตพอร์ตนี้
- setRouter() ใช้บอกว่าเป็นพอร์ตไหนของเราเตอร์ตัวไหน
- showDialog() ใช้ในการแสดงไดอะล็อกนี้โดยจะเซตขนาดตามที่กำหนดไว้

- chkInteger() ใช้ตรวจสอบว่าสตริงที่รับเข้ามาเป็นตัวเลขหรือไม่
- chkIP() ใช้ตรวจสอบความถูกต้องของไอพีแอดเดรส
- chkSubnet() ใช้ตรวจสอบความถูกต้องของซับเนตมาร์ค
- correctIP() ใช้แก้ไขไอพีแอดเดรสให้ถูกต้อง
- isInThisVector() ตรวจสอบว่าออบเจคที่ส่งมาอยู่ในเวกเตอร์ที่ส่งมาหรือไม่
- ok_actionPerformed() เป็น method ที่ใช้ในการเปลี่ยนแปลงค่าตามที่ทำการเซตในหน้าจอ GUI
- cancel_actionPerformed()เป็น method ที่ใช้ยกเลิกการเปลี่ยนแปลงค่าต่างๆ
- ส่วน method อื่นๆจะใช้สำหรับการทำงานต่างๆบนหน้าจอ GUI

2. พอร์ตซับอินเทอร์เฟส เหมือนกับการคอนฟิกพอร์ตอีเทอร์เนต แต่เพิ่มการกำหนดว่าเป็นแลนเสมือนอะไรและใช้โพรโตคอลอะไร

3. พอร์ตซีเรียล เหมือนกับการคอนฟิกพอร์ตอีเทอร์เนต แต่เพิ่มการกำหนดคล๊อกเรตเข้าไป

10.4.2 การกำหนดเราติ้งโพรโตคอล

ใช้กำหนดเราติ้งโปรโตคอลโดยจะมีให้เลือกเป็น สแตติกเราท์ ไดนามิกเราท์ อาร์ไอพี ไอจีอาร์พี โดยสามารถเพิ่มเราติ้งโปรโตคอลได้ตามที่มีให้เลือกโดยการทำงานจะอยู่ในคลาส RoutingProtocolDialog โดยมี method การทำงานดังนี้

- RoutingProtocolDialog() เป็นคอนสตรัคเตอร์ของคลาสสำหรับสร้างค่าเริ่มต้นต่างๆของไดอะล็อก
- SetList1() ใช้แสดงว่ามีเราติ้งโพรโตคอลใดบ้างที่ถูกสร้างขึ้น
- setRouter() ใช้บอกว่าเป็นพอร์ตไหนของเราเตอร์ตัวไหน
- showDialog()ใช้ในการแสดงไดอะล็อกนี้โดยจะเซตขนาดตามที่กำหนดไว้
- add_actionPerformed() เป็น method ที่ใช้ในการเพิ่ม Routing Protocol
- edit_actionPerformed() เป็น method ที่ใช้ในการแก้ไข ปรับแต่งค่าของ Routing Protocol
- remove_actionPerformed() เป็น method ที่ใช้ในการลบ Routing Protocol ที่ไม่ต้องการ
- ok_actionPerformed() เป็น method ที่ใช้ในการตกลงการเปลี่ยนแปลงค่าที่ได้กระทำ
- cancel_actionPerformed()เป็น method ที่ใช้ในการยกเลิกการเปลี่ยนแปลงทั้งหมด

10.4.3 การกำหนดการใช้งานผ่านเทลเน็ต

ใช้ในการกำหนดการใช้งานผ่านเทลเน็ต การตั้งค่ารหัสผ่าน รวมถึงการกำหนดแอกเซสคอนโทรลลิสต์ โดยการทำงานจะอยู่ในคลาส TelnetRouter โดยมี method การทำงานดังนี้

- TelnetRouter() เป็นคอนสตรัคเตอร์ของคลาสสำหรับสร้างค่าเริ่มต้นต่างๆของไดอะล็อก
- setRouter() ใช้บอกว่าเป็นเราเตอร์ตัวไหน
- showDialog()ใช้ในการแสดงไดอะล็อกนี้โดยจะเซตขนาดตามที่กำหนดไว้
- ok_actionPerformed() เป็น method ที่ใช้ในการตกลงการเปลี่ยนแปลงค่าที่ได้กระทำ

- cancel_actionPerformed() เป็น method ที่ใช้ในการยกเลิกการเปลี่ยนแปลงทั้งหมด

10.4.4 การกำหนดการทำงานผ่านทางคอนโซล

ใช้ในการกำหนดการใช้งานผ่านางคอนโซล การตั้งค่ารหัสผ่าน รวมถึงการกำหนดแอกเซสคอนโทรลลิสต์ โดยการทำงานจะอยู่ในคลาส ConsoleRouter โดยมี method การทำงานดังนี้

- ConsoleRouter() เป็นคอนสตรัคเตอร์ของคลาสสำหรับสร้างค่าเริ่มต้นต่างๆของไดอะล็อก
- setRouter() ใช้บอกว่าเป็นเราเตอร์ตัวไหน
- showDialog()ใช้ในการแสดงไดอะล็อกนี้โดยจะเซตขนาดตามที่กำหนดไว้
- ok_actionPerformed() เป็น method ที่ใช้ในการตกลงการเปลี่ยนแปลงค่าที่ได้กระทำ
- cancel_actionPerformed() เป็น method ที่ใช้ในการยกเลิกการเปลี่ยนแปลงทั้งหมด

10.4.5 การสร้างแอกเซสคอนโทรลลิสต์

เป็นการสร้างแอกเซสคอนโทรลลิสต์ผ่านทางหน้าจอโดยสามารถแก้ไขเพิ่มเติมได้ผ่านทางหน้าจอนี้โดยในส่วน method การทำงานจะอยู่ในคลาส ACLDialog เช่นเดียวกันกับสวิตช์

10.4.6 ในส่วนการสร้างConfiguration File

ในส่วนนี้เมื่อมีการเรียกใช้จะเรียก method genConfigFile()ในคลาสเราเตอร์มาใช้ โดย method นี้จะสร้างไฟล์ใหม่ขึ้นมาให้เราเซฟConfiguration Fileที่ได้ไปไ

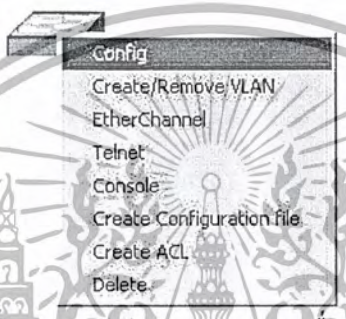
บทที่ 11
ตัวอย่างและการทดสอบการทำงานของโปรแกรม

11.1 ตัวอย่างทดสอบการตั้งค่าสวิตช์

11.1.1 ทดสอบการกำหนดรายละเอียดทั่วไปของสวิตช์

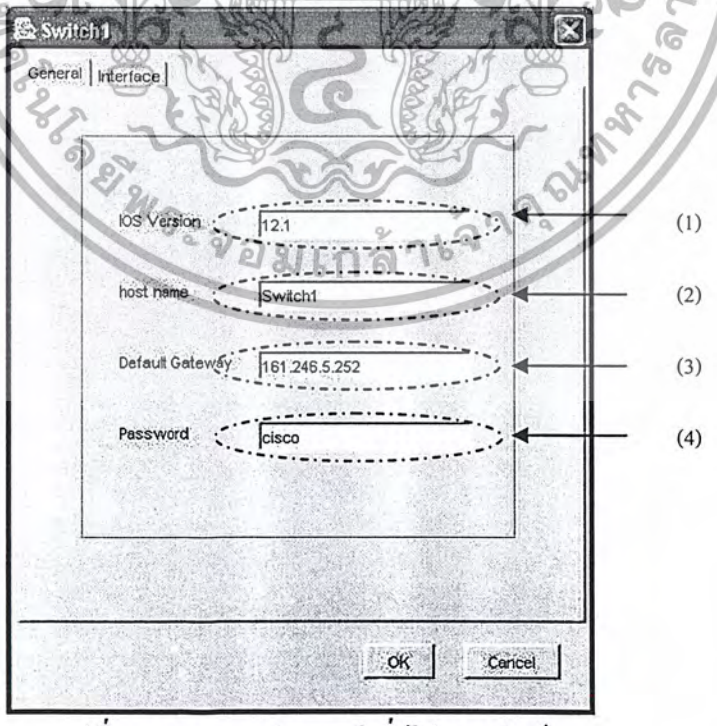
ตัวอย่างนี้เป็นการสร้างค่าติดตั้งเพื่อกำหนดค่า IOS Version, host name, default gateway และ password ให้กับสวิตช์ ผลลัพธ์ที่ได้คือ Config-File I ในภาคผนวก ก มีขั้นตอนการตั้งค่าสวิตช์ดังนี้

ขั้นที่1 เปิดโปรแกรม สร้างสวิตช์โดยเลือกรุ่นที่ต้องการ และเลือกเมนู Config ดังรูปที่ 11-1



รูปที่ 11-1 แสดงขั้นตอนการใช้งานเมนู Configเพื่อกำหนดค่าให้สวิตช์

ขั้นที่2 กำหนดค่า IOS Version, hostname, Default Gateway และ Password ดังรูปที่ 11-2

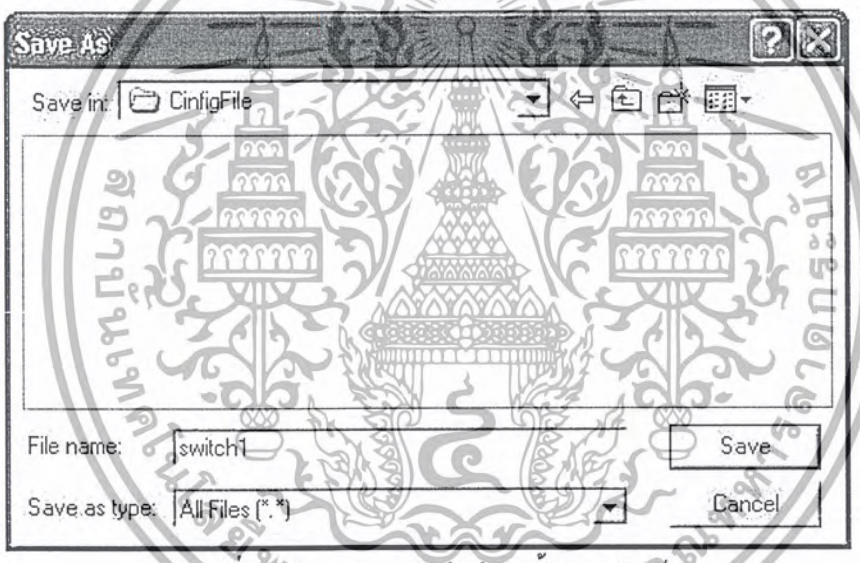


รูปที่ 11-2 แสดงการกำหนดค่าทั่วไปของสวิตช์

ขั้นที่3 สร้าง configuration file โดยเลือกที่เมนู Create Configuration File ดังรูปที่ 11-3

รูปที่ 11-3 แสดงการใช้งานเมนู Create Configuration File เพื่อสร้างค่าติดตั้งของสวิตช์

ขั้นที่4 เลือก directory ที่ต้องการเก็บไฟล์และตั้งชื่อไฟล์ที่สร้างขึ้น ในที่นี้ชื่อ switch1 ดังรูปที่ 11-4



รูปที่ 11-4 แสดงการบันทึกค่าติดตั้งของสวิตช์

Configuration File ที่ได้จากโปรแกรมเป็นไฟล์ที่มีนามสกุลเป็น .text ดังนั้นสามารถเปิดได้จากโปรแกรมสำหรับสร้างเอกสารทั่วไป จากการตั้งค่าข้างต้นจะได้ Configuration File ดังในภาคผนวก ก Config-file I ในที่นี้จะขอแสดงเฉพาะในส่วนที่การตั้งค่าหรือมีการเปลี่ยนแปลงค่าเท่านั้น จากตัวอย่างสามารถเปรียบเทียบค่าติดตั้งในส่วนต่างๆได้ดังตัวเลขที่กำกับไว้ในวงเล็บ ดังรูปที่ 11-5

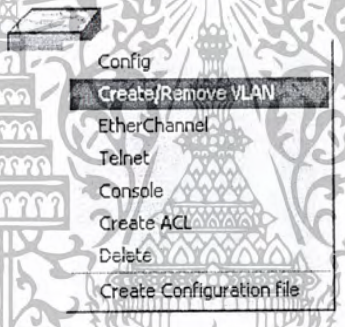
```
version 12.1                          .....(1)
...
!
hostname Switch1                      .....(2)
enable password cisco                 .....(4)
...
ip default-gateway 161.246.5.252      .....(3)
```

รูปที่ 11-5 แสดงค่าติดตั้งบางส่วนจากภาคผนวก ก Config-File I

11.1.2 ทดสอบการกำหนดแลนเสมือน

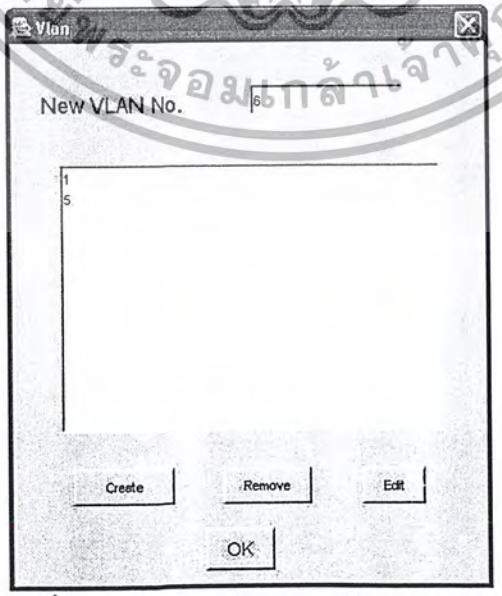
ตัวอย่างนี้เป็นตัวอย่างการสร้างค่าติดตั้งที่มีการสร้างแลนเสมือน 5, 6 และ 7 และกำหนดค่าไอพีแอดเดรสให้แก่สวิตช์ ผลลัพธ์ที่ได้คือ Config-File II ในภาคผนวก ก มีขั้นตอนการตั้งค่าสวิตช์ดังนี้

ขั้นที่1 เลือกเมนู Create/Remove VLAN ดังรูปที่ 11-6



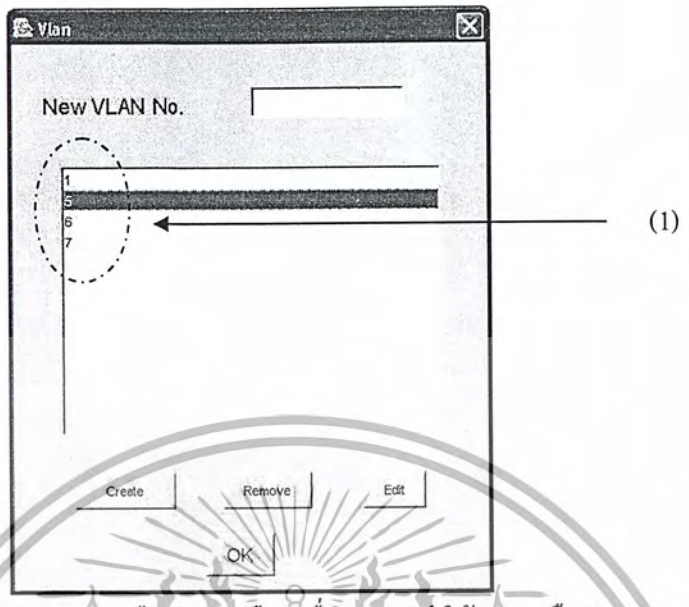
รูปที่ 11-6 แสดงการใช้งานเมนู Create/Remove VLAN

ขั้นที่2 กรอกหมายเลขแลนเสมือนแล้วกดปุ่ม Add ดังรูปที่ 11-7



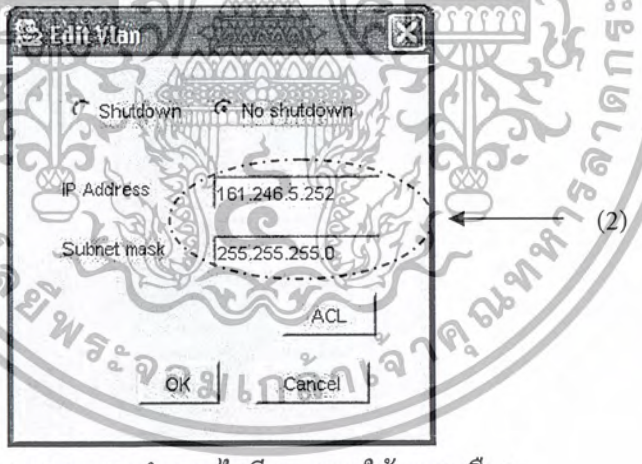
รูปที่ 11-7 แสดงการสร้างแลนเสมือน 5, 6 และ7

ขั้นที่3 กำหนดไอพีแอดเดรสให้แลนเสมือนโดยเลือกแลนเสมือน 5 แล้วเลือกปุ่ม Edit ดังรูปที่ 11-8



รูปที่ 11-8 แสดงการเลือกแลนเสมือนเพื่อกำหนดค่าให้แลนเสมือน

ขั้นที่4 กำหนดไอพีแอดเดรสและ Subnet เลือกปุ่ม OK รูปที่ 11-9



รูปที่ 11-9 แสดงการกำหนดไอพีแอดเดรสให้แลนเสมือน

ขั้นที่5 สร้าง Configuration File และสังเกตผลที่ได้ ดังรูปที่ 11-10

```
86>     !
87>     interface Vlan1                                    ..... (1)
88>     no ip address
89>     no ip route-cache
90>     !
```

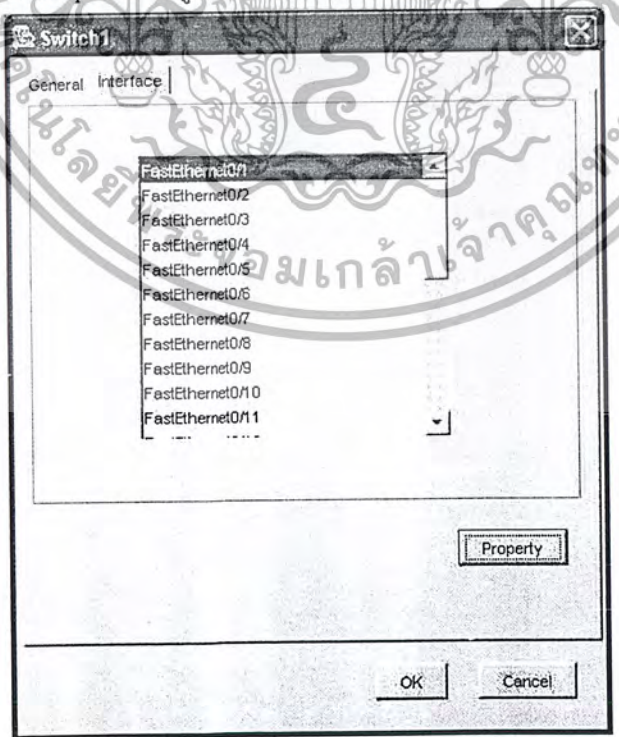
```
91>     interface Vlan5                                    .....(1)
92>     ip address 161.246.5.252 255.255.255.0             .....(1)
93>     no ip route-cache
94>     !
95>     interface Vlan6                                    .....(2)
96>     no ip address
97>     no ip route-cache
98>     !
99>     interface Vlan7                                    .....(1)
100>    no ip address
101>    no ip route-cache
102>    !
```

รูปที่ 11-10 แสดงค่าติดตั้งบางส่วนจากภาคผนวก ค Config-File II

11.1.3 ทดสอบการกำหนดค่าบนอินเทอร์เฟสของสวิตช์

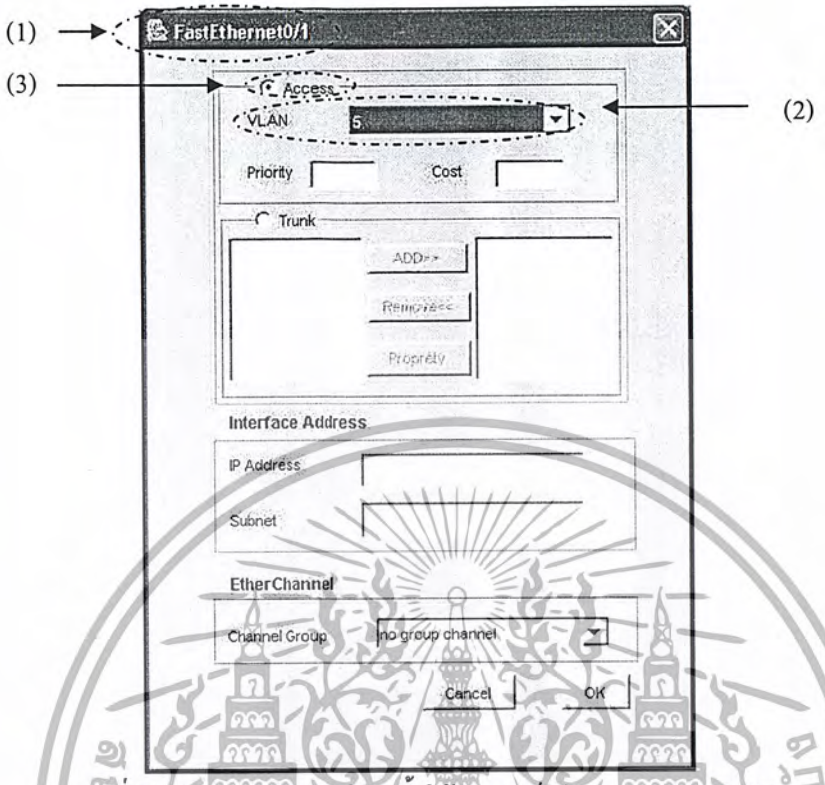
ตัวอย่างนี้เป็นการสร้างค่าติดตั้งเพื่อกำหนดค่าให้อินเทอร์เฟส FastEthernet0/1 มีโหมดการทำงานเป็นแอคเซสลิงค์ และเป็นสมาชิกของแลนเสมือน 5 ผลลัพธ์ที่ได้คือ Config-File III ในภาคผนวก ก มีขั้นตอนการตั้งค่าสวิตช์ดังนี้

ขั้นที่1 เลือกเมนู Config เลือกแถบ Interface เลือก FastEthernet0/1 เพื่อตั้งค่าบนอินเทอร์เฟส FastEtherNet0/1 และเลือก Properties ดังรูปที่ 11-11



รูปที่ 11-11 แสดงการเลือกอินเทอร์เฟสของสวิตช์เพื่อกำหนดค่าติดตั้ง

ขั้นที่2 กำหนดโหมดการทำงานเป็นแอคเซสลิงค์ และเป็นสมาชิกของแลนเสมือน 5 ดังรูปที่ 11-12



รูปที่ 11-12 แสดงกำหนดค่าติดตั้งให้อินเทอร์เฟส FastEthernet0/1

ขั้นที่3 สร้าง Configuration File และสังเกตผลที่ได้ ดังรูปที่ 11-13

```
interface FastEthernet0/1 ....(1)
switchport acces vlan 5 ....(2)
switchport mode access ....(3)
no ip address
```

รูปที่ 11-13 แสดงค่าติดตั้งบางส่วนจากภาคผนวก ก Config-File III

11.1.4 ทดสอบการกำหนดพอร์ตแชนแนล

ตัวอย่างนี้เป็นการกำหนดพอร์ตแชนแนล 2 แชนแนล โดยแต่ละแชนแนลมีโหมดการทำงานเป็นโหมดทรังค์ลิงค์ ประกอบด้วยแลนเสมือน 5, 6 และ 7 แต่ละแชนแนลมีการกำหนดค่า Priority ดังนี้

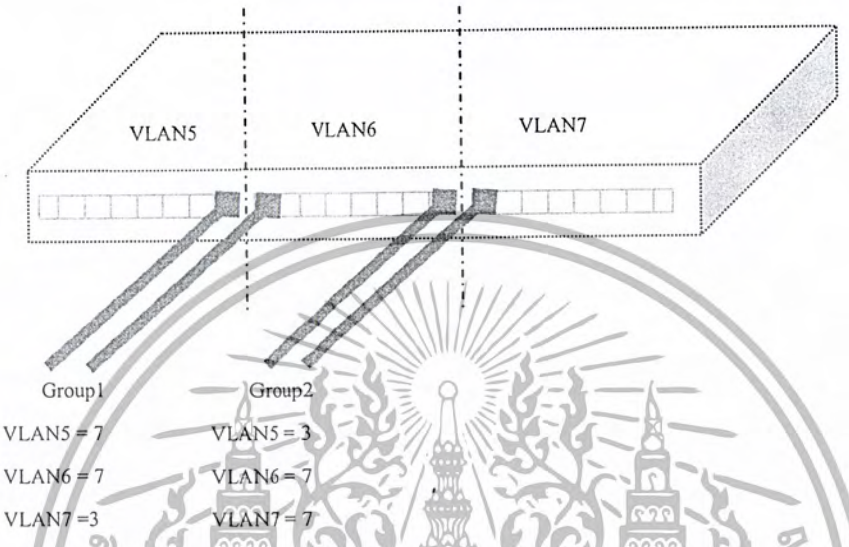
Group1

- แลนเสมือน 5 มีค่า Priority เท่ากับ 7
- แลนเสมือน 6 มีค่า Priority เท่ากับ 7
- แลนเสมือน 7 มีค่า Priority เท่ากับ 3

Group2

- แลนเสมือน 5 มีค่า Priority เท่ากับ 3
- แลนเสมือน 6 มีค่า Priority เท่ากับ 7
- แลนเสมือน 7 มีค่า Priority เท่ากับ 7

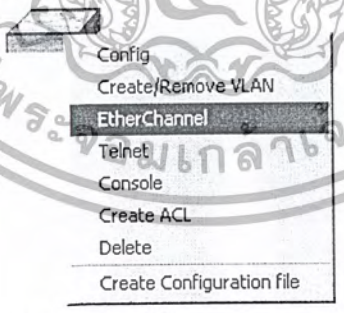
จากตัวอย่างข้างต้น สามารถแสดงได้ดังรูปที่ 11-14



รูปที่ 11-14 แสดงตัวอย่างการแบ่งการใช้งานแลนเสมือนบนสวิตช์

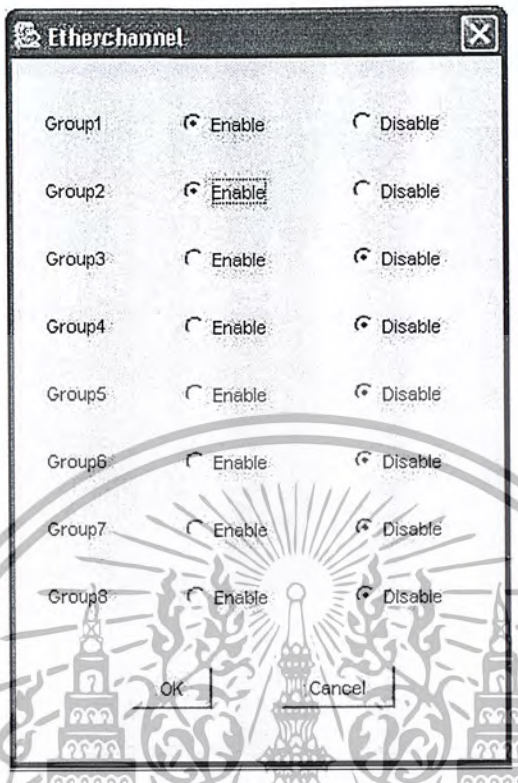
ผลลัพธ์ที่ได้คือ Config-File IV ในภาคผนวก ก มีขั้นตอนการตั้งค่าสวิตช์ดังนี้

ขั้นที่1 เลือกเมนู Ether Channel รูปที่ 11-15



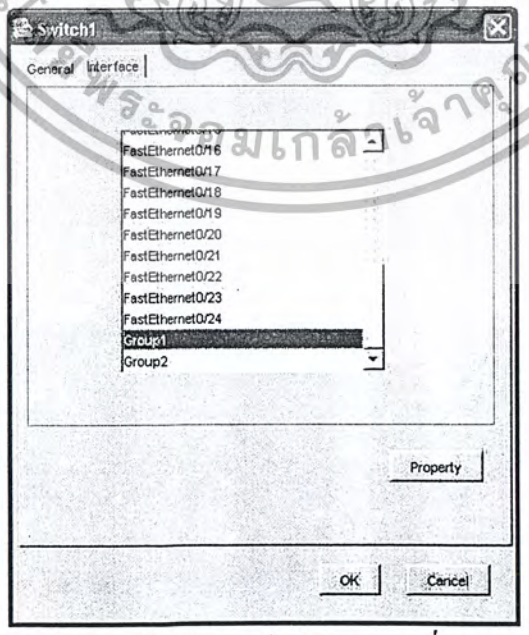
รูปที่ 11-15 แสดงการใช้งานเมนู EtherChannel

ขั้นที่2 เลือก Enable กรุ๊ปที่ต้องการ จากตัวอย่างเป็นการสร้างพอร์ตแชนแนล ดังรูปที่ 11-16



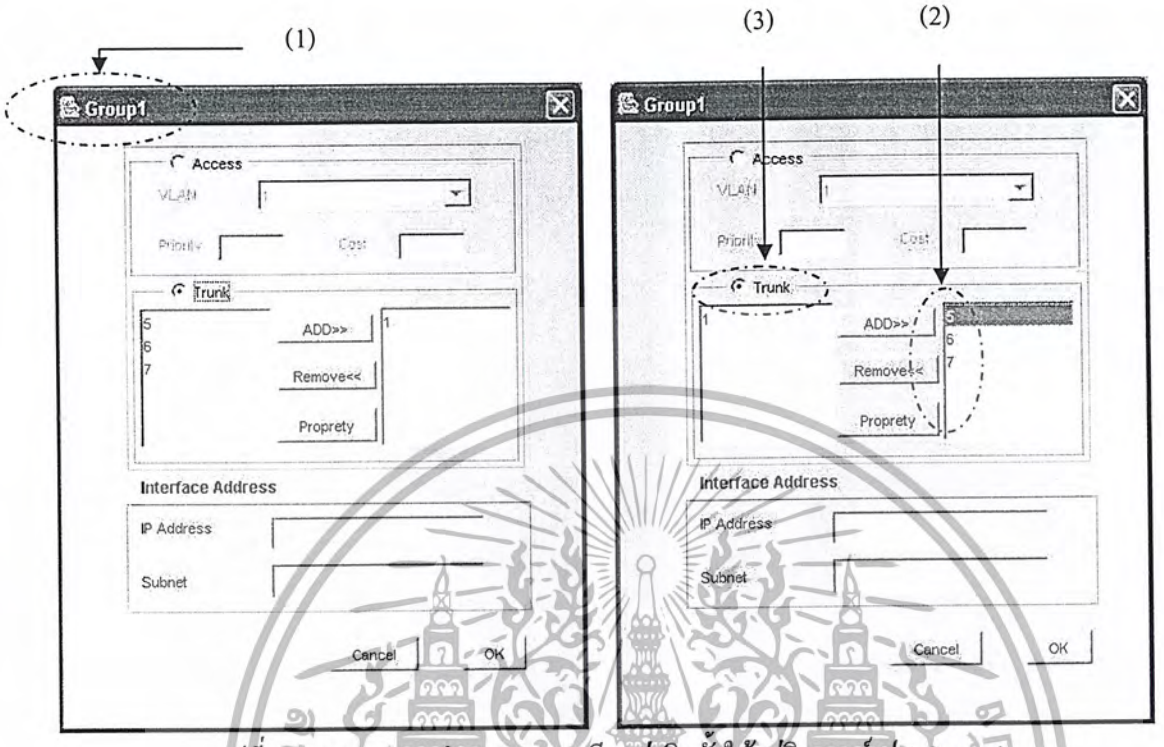
รูปที่ 11-16 แสดงการสร้างพอร์ตแชนแนล Group1 และ Group2

ขั้นที่3 เป็นการกำหนดรายละเอียดให้กับพอร์ตแชนแนล Group1 และ Group2 โดยกำหนดให้รายละเอียดดังตัวอย่าง โดยเลือกเมนู Config, เลือกแถบ Interface, เลือก Group1 และเลือก Properties ดังรูปที่ 11-17



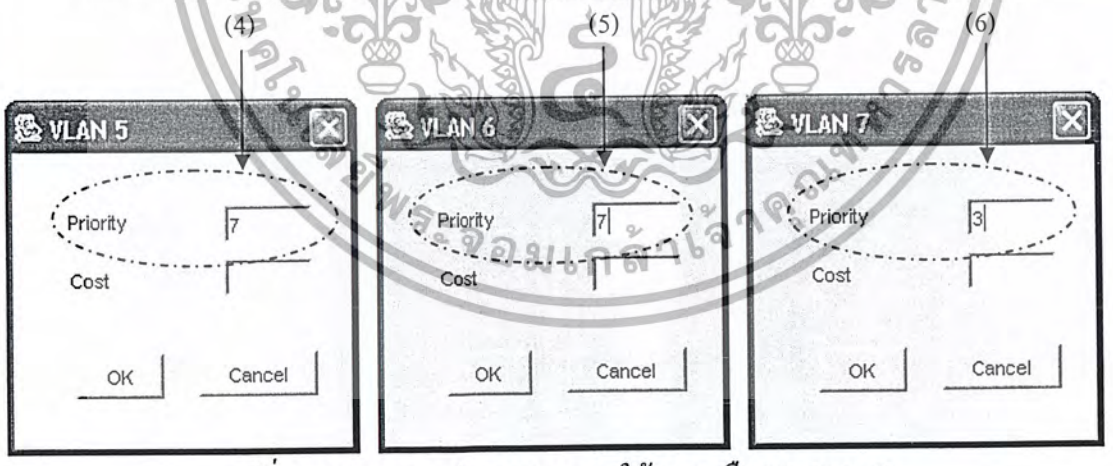
รูปที่ 11-17 แสดงการเลือกอินเทอร์เฟส Group1 เพื่อกำหนดค่าติดตั้ง

ขั้นที่4 เป็นการกำหนดโหมดการทำงานและสมาชิกแลนเสมือนให้กับพอร์ตแชนแนล Group1 ดังรูปที่ 11-18



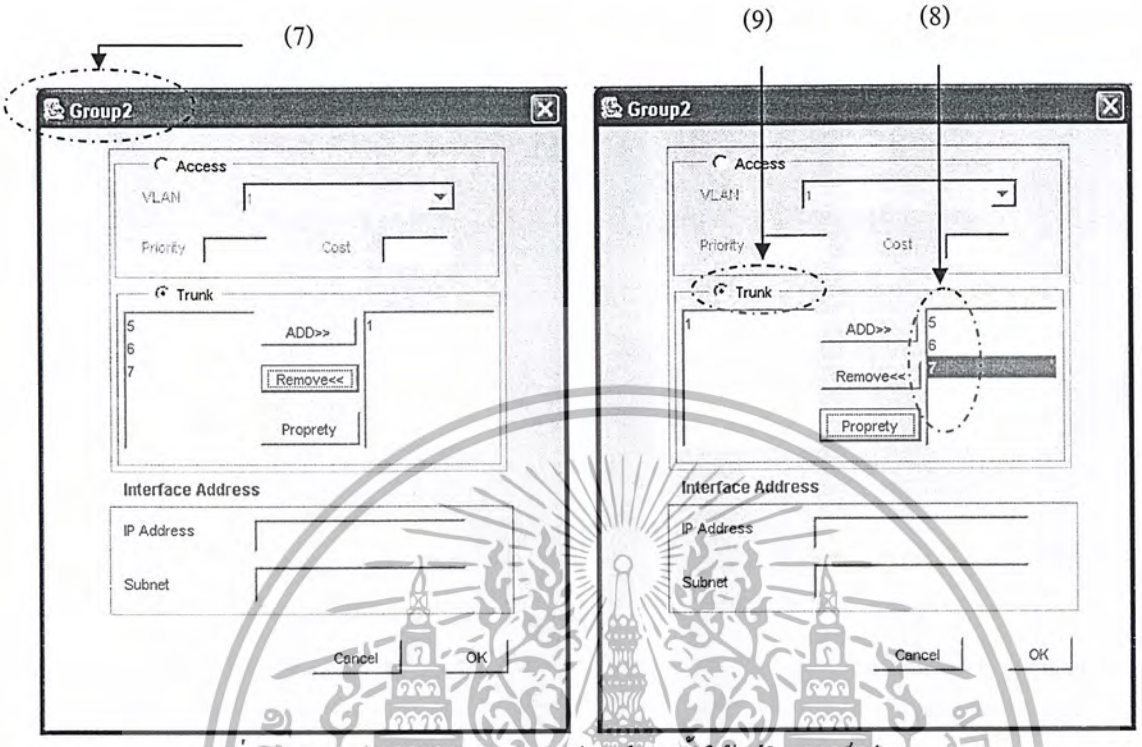
รูปที่ 11-18 แสดงการกำหนดรายละเอียดค่าติดตั้งให้แก่อินเทอร์เฟส Group1

ขั้นที่5 กำหนดค่า Priority ให้กับแต่ละแลนเสมือนในพอร์ตแชนแนล Group1 ดังรูปที่ 11-19



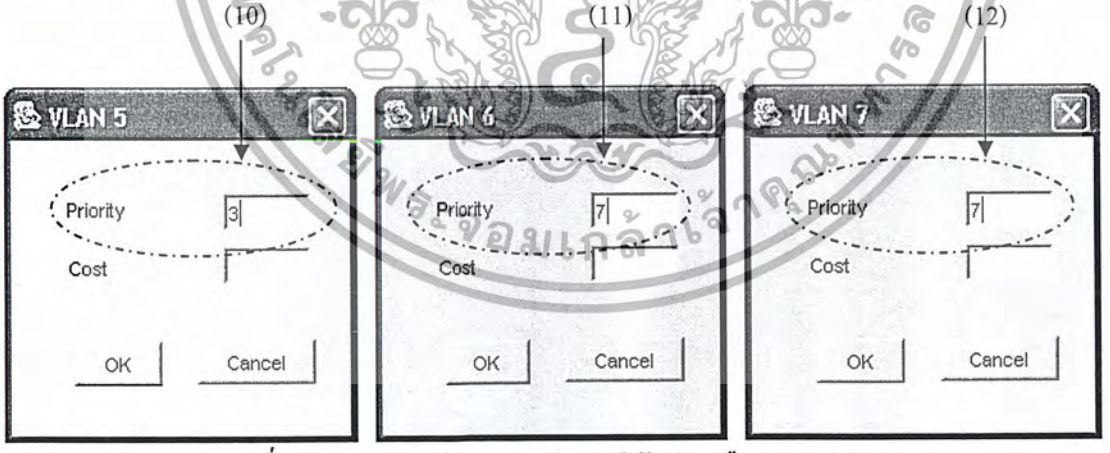
รูปที่ 11-19 แสดงการกำหนด Priority ให้แลนเสมือน 5, 6 และ 7

ขั้นที่6 เป็นการกำหนดโหมดการทำงานและสมาชิกแลนเสมือนให้กับพอร์ตแชนแนล Group2 ดังรูปที่ 11-20



รูปที่ 11-20 แสดงการกำหนดรายละเอียดค่าติดตั้งให้แก่อินเทอร์เฟส Group2

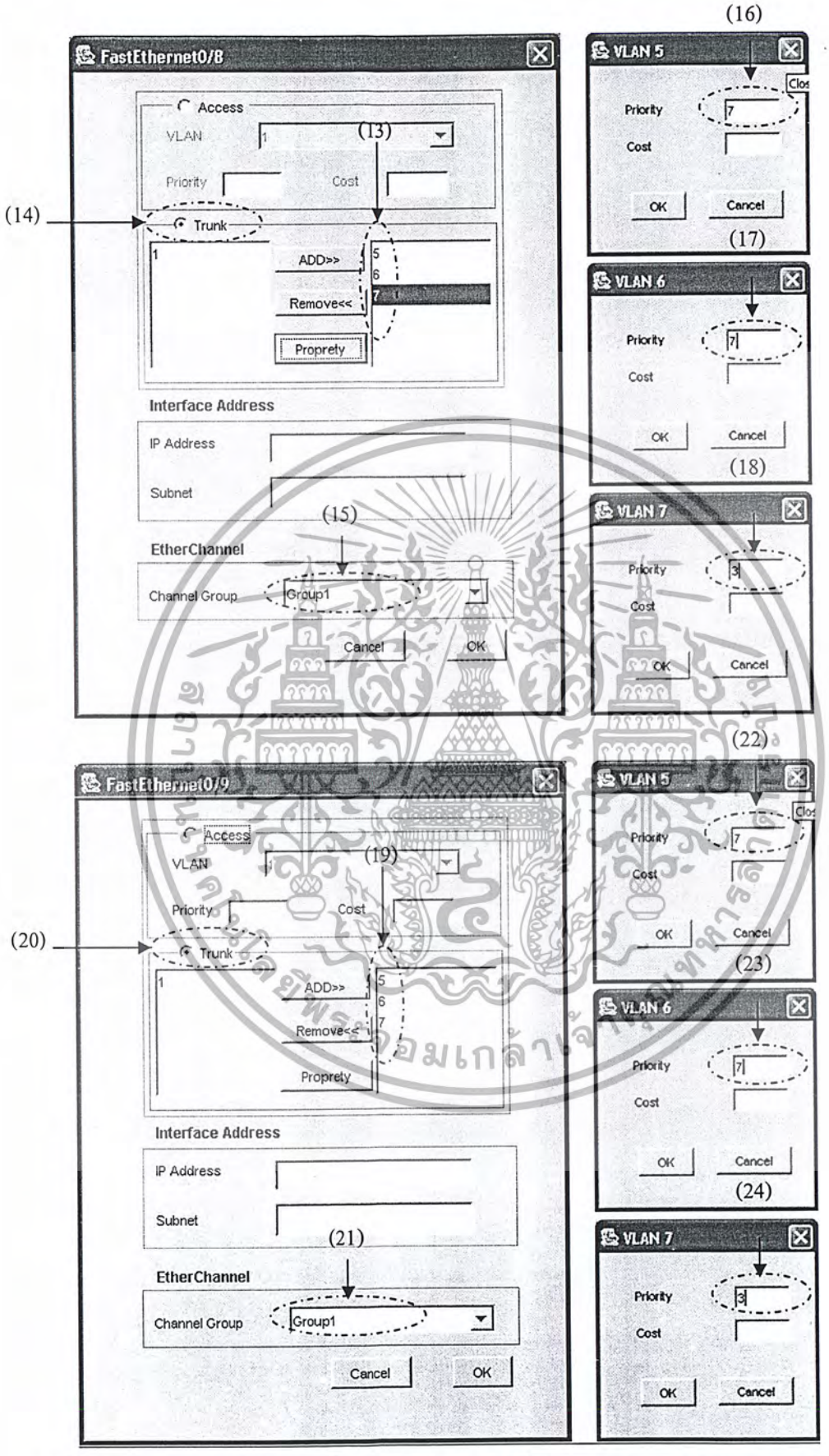
ขั้นที่7 กำหนดค่า Priority ให้กับแต่ละแลนเสมือนในพอร์ตแชนแนล Group2 ดังรูปที่ 11-21



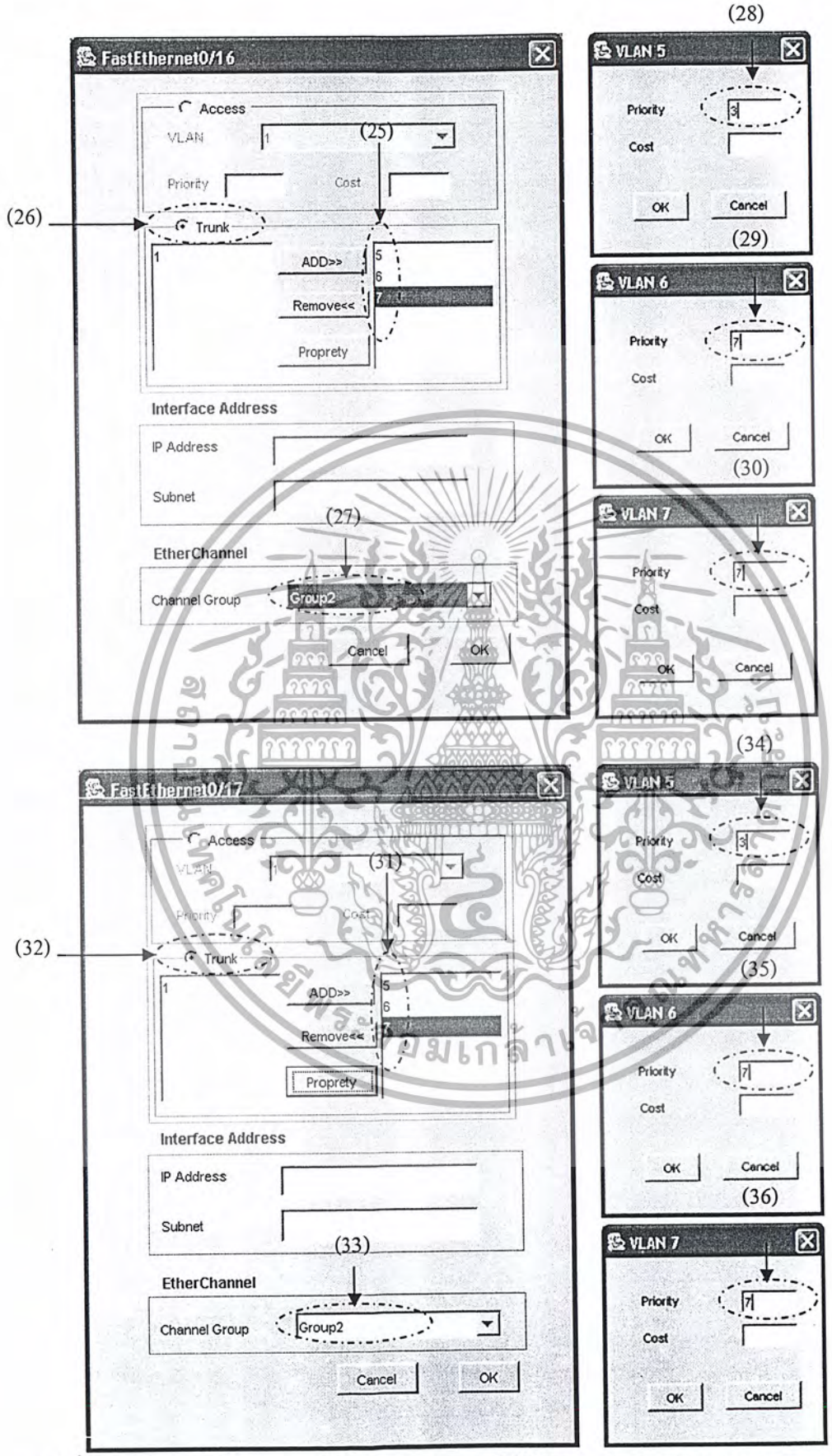
รูปที่ 11-21 แสดงการกำหนด Priority ให้แลนเสมือน 5, 6 และ 7

ขั้นที่8 การกำหนดพอร์ตแชนแนล Group1 ให้กับอินเทอร์เฟสของสวิตช์ ดังรูปที่ 11-22

ขั้นที่9 การกำหนดพอร์ตแชนแนล Group2 ให้กับอินเทอร์เฟสของสวิตช์ ดังรูปที่ 11-23



รูปที่ 11-22 แสดงการกำหนดพอร์ตแชนแนล Group1 ให้กับ FastEthernet0/8, FastEthernet0/9



รูปที่ 11-23 แสดงการกำหนดพอร์ตแชนแนล Group2 ให้กับ FastEthernet0/16, FastEthernet0/17

ขั้นที่10 สร้าง Configuration File และสังเกตผลที่ได้ ดังรูปที่ 11-23

```
15>  interface Port-channel1                          .....(1)
16>  switchport trunk allowed vlan 5,6,7              .....(2)
17>  switchport mode trunk                           .....(3)
18>  no ip address
19>  flowcontrol send off
20>  spanning-tree vlan 5 port-priority 7            .....(4)
21>  spanning-tree vlan 6 port-priority 7            .....(5)
22>  spanning-tree vlan 7 port-priority 3            .....(6)
23>  !
24>  interface Port-channel2                         .....(7)
25>  switchport trunk allowed vlan 5,6,7             .....(8)
26>  switchport mode trunk                           .....(9)
27>  no ip address
28>  flowcontrol send off
29>  spanning-tree vlan 5 port-priority 3            .....(10)
30>  spanning-tree vlan 6 port-priority 7            .....(11)
31>  spanning-tree vlan 7 port-priority 7            .....(12)
32>
58>  interface FastEthernet0/8
59>  switchport trunk allowed vlan 5,6,7             .....(13)
60>  switchport mode trunk                           .....(14)
61>  no ip address
62>  channel-group 1 mode on                         .....(15)
63>  spanning-tree vlan 5 port-priority 7            .....(16)
64>  spanning-tree vlan 6 port-priority 7            .....(17)
65>  spanning-tree vlan 7 port-priority 3            .....(18)
66>  !
67>  interface FastEthernet0/9
68>  switchport trunk allowed vlan 5,6,7             .....(19)
69>  switchport mode trunk                           .....(20)
70>  no ip address
71>  channel-group 1 mode on                         .....(21)
72>  spanning-tree vlan 5 port-priority 7            .....(22)
73>  spanning-tree vlan 6 port-priority 7            .....(23)
74>  spanning-tree vlan 7 port-priority 3            .....(24)
```

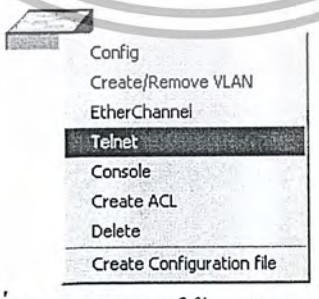
```
>  ...
94>   interface FastEthernet0/16
95>   switchport trunk allowed vlan 5,6,7          .....(25)
96>   switchport mode trunk                        .....(26)
97>   no ip address
98>   channel-group 2 mode on                      .....(27)
99>   spanning-tree vlan 5 port-priority 3         .....(28)
100>  spanning-tree vlan 6 port-priority 7         .....(29)
101>  spanning-tree vlan 7 port-priority 7         .....(30)
102>  !
103>  interface FastEthernet0/17
104>  switchport trunk allowed vlan 5,6,7          .....(31)
105>  switchport mode trunk                        .....(32)
106>  no ip address
107>  channel-group 2 mode on                      .....(33)
108>  spanning-tree vlan 5 port-priority 3         .....(34)
109>  spanning-tree vlan 6 port-priority 7         .....(35)
110>  spanning-tree vlan 7 port-priority 7         .....(36)
```

รูปที่ 11-24 แสดงค่าติดตั้งบางส่วนจากภาคผนวก ก Config-File IV

11.1.5 ทดสอบการกำหนดรหัสผ่านการเข้าใช้งานผ่านโปรแกรมเทลเน็ต

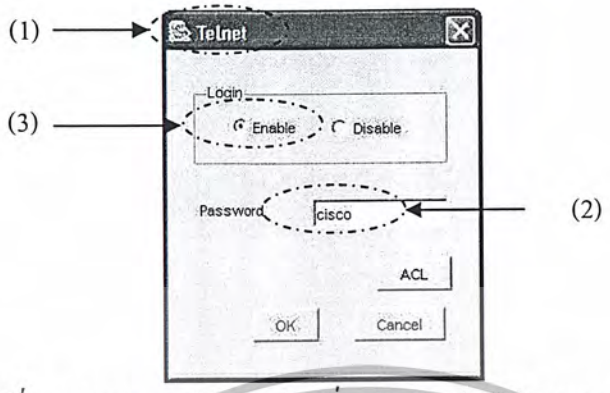
ตัวอย่างนี้เป็นการสร้างค่าติดตั้งเพื่อกำหนดรหัสผ่านการเข้าใช้งานสวิตช์หรือเราเตอร์ผ่านโปรแกรมเทลเน็ต โดยกำหนดรหัสผ่านเป็น "cisco" ผลลัพธ์ที่ได้คือ Config-File V ในภาคผนวก ก มีขั้นตอนการตั้งค่าดังนี้

ขั้นที่1 เลือกเมนู Telnet ดังรูปที่ 11-24



รูปที่ 11-25 แสดงการใช้งานเมนู Telnet

ขั้นที่2 เลือก Enable เมื่อต้องการให้มีการใช้รหัสผ่านในการล็อกอินโดยการใช้โปรแกรมเทลเน็ต และกรอกรหัสผ่านดังรูปที่ 11-25



รูปที่ 11-26 แสดงการใช้รหัสผ่านเมื่อรล็อกอินผ่านโปรแกรมเทลเน็ต

ขั้นที่3 สร้าง Configuration File และสังเกตผลที่ได้ ดังรูปที่ 11-26

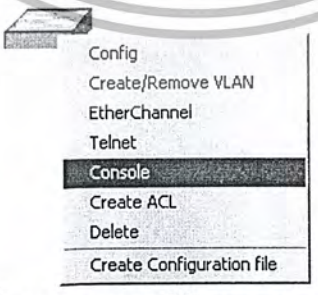
```
line vty 0 15
 password cisco
 login
```

รูปที่ 11-27 *แสดงค่าติดตั้งบางส่วนจากภาคผนวก ก Config-File VI*

11.1.6 ทดสอบการกำหนดรหัสผ่านการเข้าใช้งานผ่านพอร์ตคอนโซล

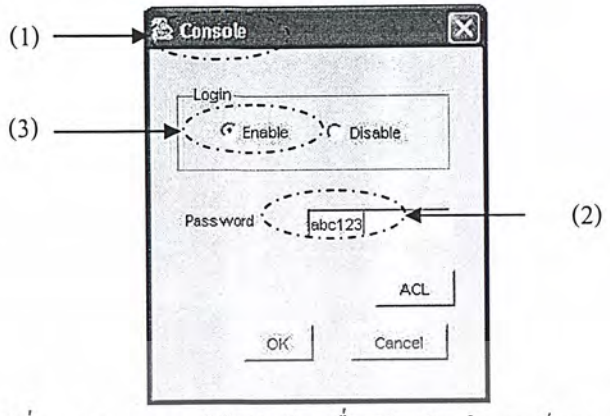
ตัวอย่างนี้เป็นตัวอย่างการสร้างค่าติดตั้งเพื่อกำหนดรหัสผ่านการเข้าใช้งานสวิตช์ หรือเราเตอร์ผ่านพอร์ตคอนโซล โดยกำหนดรหัสผ่านเป็น "abc123" ผลลัพธ์ที่ได้คือ Config-File VI ในภาคผนวก ก มีขั้นตอนการตั้งค่าดังนี้

ขั้นที่1 เลือกเมนู Console ดังรูปที่ 11-27



รูปที่ 11-28 แสดงการใช้งานเมนู Console

ขั้นที่2 เลือก Enable เมื่อต้องการให้มีการใช้รหัสผ่านในการล็อกอินพอร์ตคอนโซล และกรอกรหัสผ่านดังรูปที่ 11-28



รูปที่ 11-29 แสดงการใช้รหัสผ่านเมื่อรล็อกอินผ่านพอร์ตคอนโซล

ขั้นที่3 สร้าง Configuration File และสังเกตผลที่ได้ ดังรูปที่ 11-29

```
line con 0
password abc123
login
```

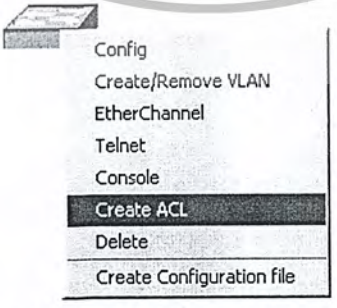
รูปที่ 11-30 แสดงค่าติดตั้งบางส่วนจากภาคผนวก ก Config-File VI

11.1.7 ทดสอบการกำหนดเอซีแอล

11.1.7.1 ตัวอย่างการกำหนด Standard ACL

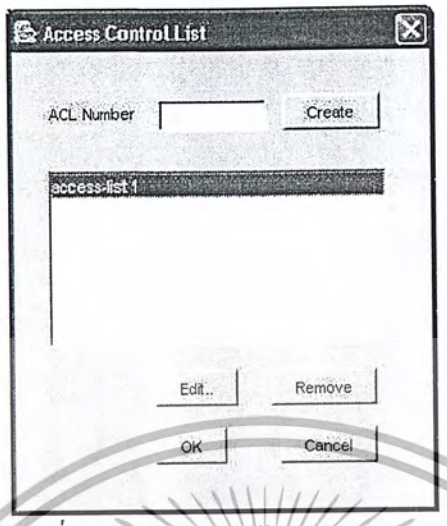
ตัวอย่างนี้เป็นตัวอย่างการสร้างค่าติดตั้งเพื่อสร้าง Standard ACL 1 โดยมีกฎว่า ให้อนุญาตแพ็กเกตจากเครือข่าย 161.246.5.0 ผ่าน และอนุญาตแพ็กเกตจากทุกเครือข่ายผ่าน ผลลัพธ์ที่ได้คือ Config-File VII ในภาคผนวก ก มีขั้นตอนการตั้งค่าดังนี้

ขั้นที่1 เลือกเมนู Create ACL ดังรูปที่ 11-30



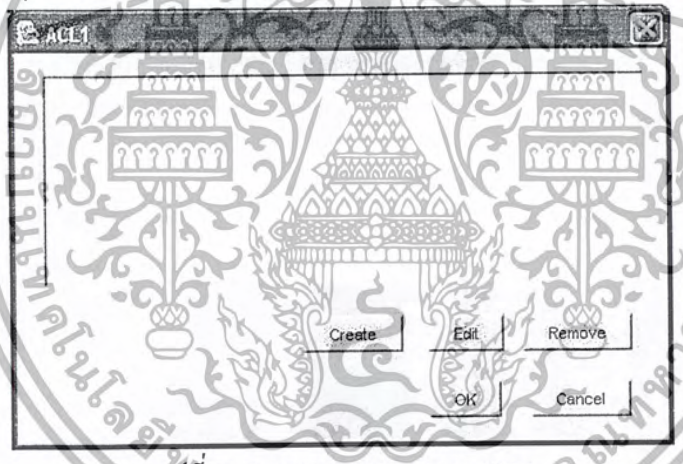
รูปที่ 11-31 แสดงการใช้งานเมนู Create ACL

ขั้นที่2 สร้างหมายเลข ACL ที่มีค่าอยู่ในช่วง 1-99 จากตัวอย่างเป็นการสร้าง Access-List 1 แล้ว เลือกปุ่ม Edit เพื่อสร้าง ACE (Access Control Entry สำหรับ Access-List 1) ดังรูปที่ 11-31



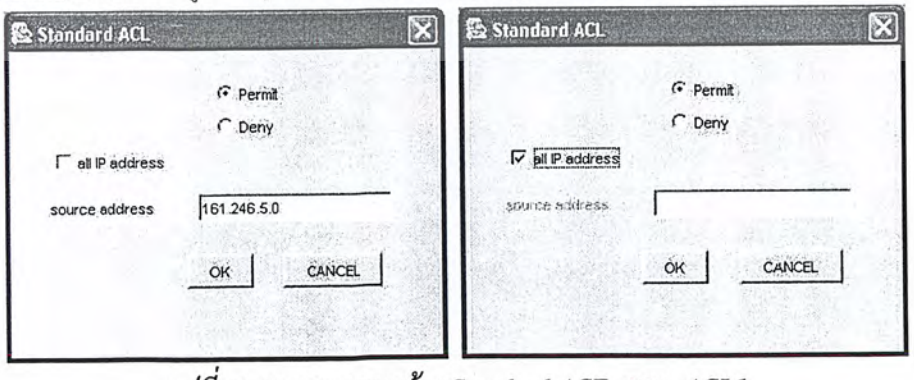
รูปที่ 11-32 แสดงการสร้าง Access-List 1

ขั้นที่3 เลือกปุ่ม Create เพื่อสร้าง ACEs ดังรูปที่ 11-32



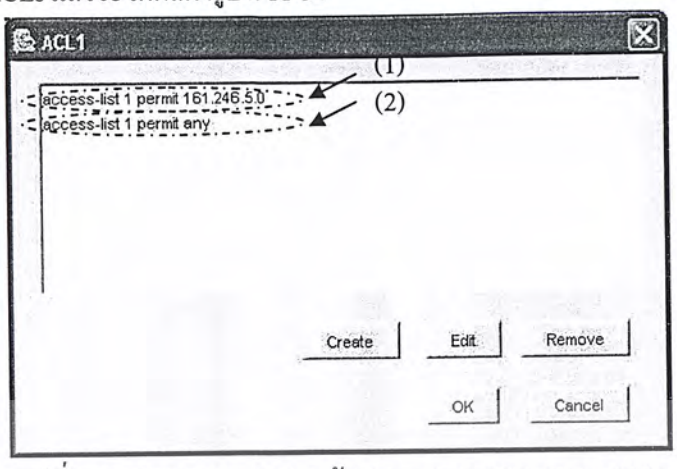
รูปที่ 11-33 แสดง ACEs ของ ACL1

ขั้นที่4 กำหนดข้อมูลการ permit และ deny ตามดังรูปที่ 11-33



รูปที่ 11-34 แสดงการสร้าง Standard ACEs ของ ACL1

เมื่อสร้าง ACEs แล้วจะได้ผลดังรูปที่ 11-34



รูปที่ 11-35 แสดงผลจากการสร้าง Standard ACEs ของ ACL1

ขั้นที่5 สร้าง Configuration File และสังเกตผลที่ได้ ดังรูปที่ 11-35

```
access-list 1 permit 161.246.5.0
access-list 1 permit any
```

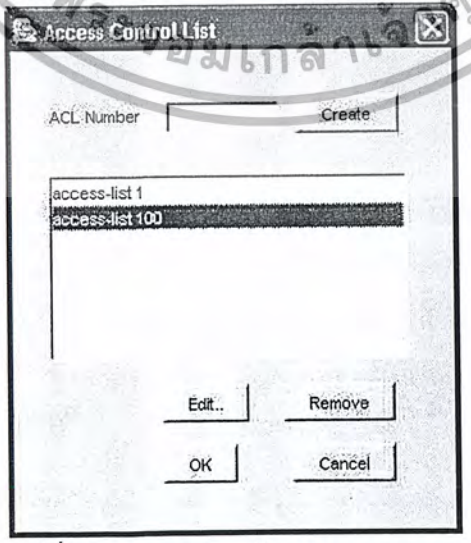
รูปที่ 11-36 แสดงค่าติดตั้งบางส่วนจากภาคผนวก ก Config-File VII

11.1.7.2 ตัวอย่างการกำหนด Extend ACL

ตัวอย่างนี้เป็นตัวอย่างการสร้างค่าติดตั้งเพื่อสร้าง Extend ACL 100 โดยมีกฎว่า ไม่อนุญาตให้แพ็กเกตที่เป็นบริการ TCP ที่มีแอดเดรสต้นทางจาก 161.246.5.0 ผ่าน และอนุญาตแพ็กเกตที่เป็นบริการ TCP จากทุกเครือข่ายผ่าน ผลลัพธ์ที่ได้คือ Config-File VIII ในภาคผนวก ก มีขั้นตอนการตั้งค่าดังนี้

ขั้นที่1 เลือกเมนู Create ACL ดังรูปที่ 11-30

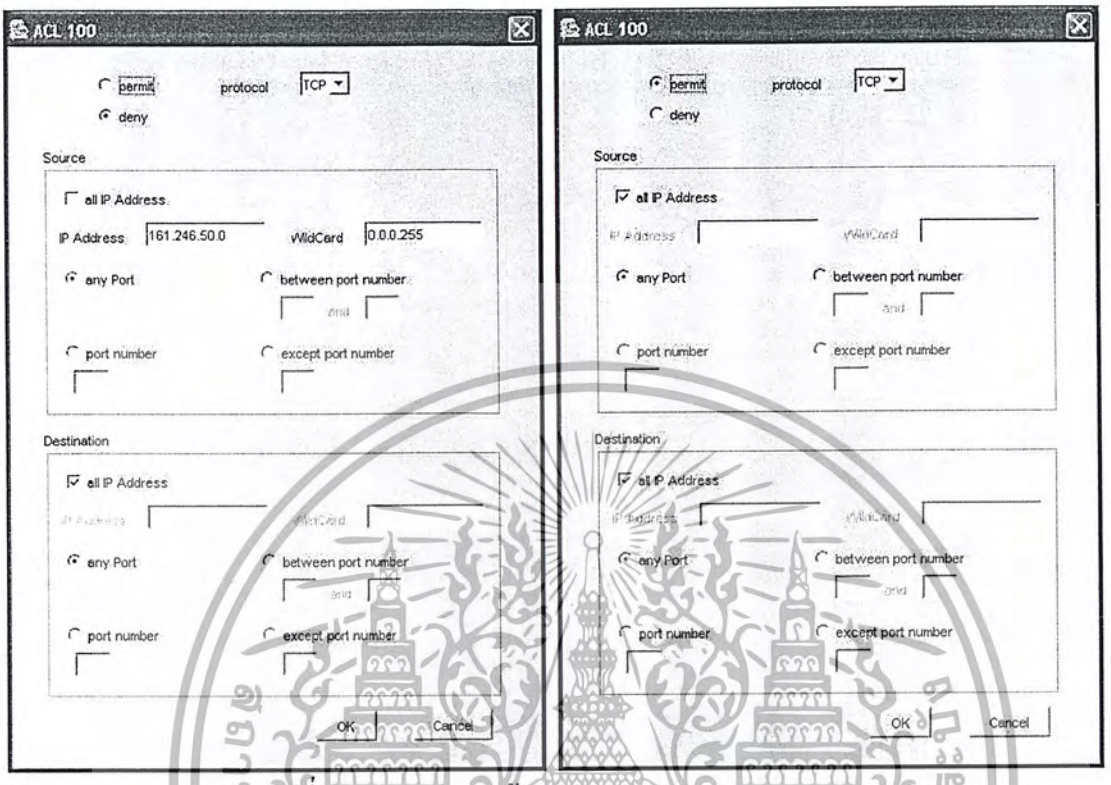
ขั้นที่2 สร้างหมายเลข ACL ที่มีค่าอยู่ในช่วง 100-199 ดังรูปที่ 11-36



รูปที่ 11-37 แสดงการสร้าง Access-List 100

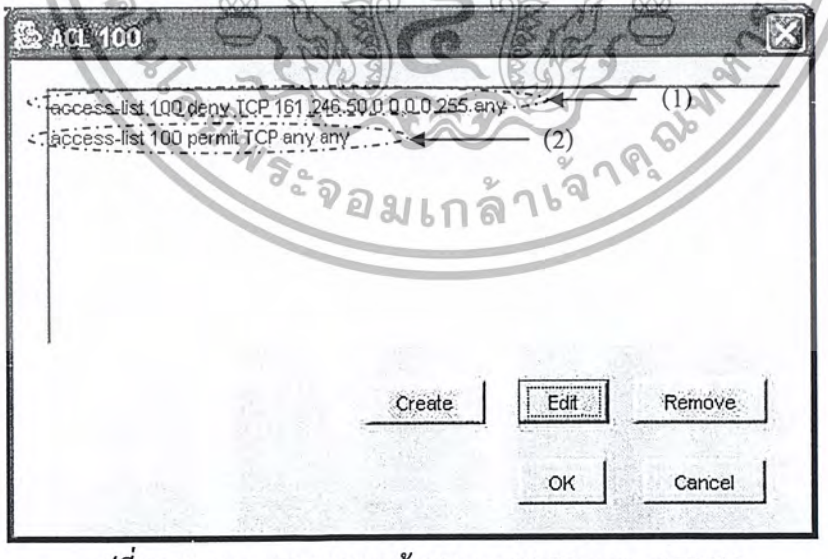
ขั้นที่3 เลือกปุ่ม Create เพื่อสร้าง ACEs

ขั้นที่4 กำหนดข้อมูลการ permit และ deny ตามดังรูปที่ 11-37



รูปที่ 11-38 แสดงการสร้าง Extend ACEs ของ ACL100

เมื่อสร้าง ACE แล้วจะได้ผลดังรูปที่ 11-38



รูปที่ 11-39 แสดงผลจากการสร้าง Extend ACEs ของ ACL100

ขั้นที่5 สร้าง Configuration File และสังเกตผลที่ได้ดังรูปที่ 11-39

```
173>    access-list 100 deny TCP 161.246.50.0 0.0.0.255 any ....(1)
174>    access-list 100 permit TCP any any ....(2)
```

รูปที่ 11-40 แสดงค่าติดตั้งบางส่วนจากภาคผนวก ก Config-File VIII

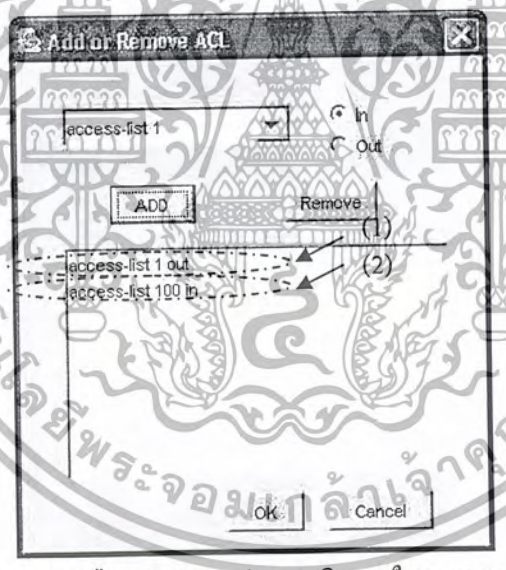
11.1.8 ทดสอบการนำเอซีแอลไปใช้งาน

ตัวอย่างนี้เป็นตัวอย่างการสร้างค่าติดตั้งเพื่อกำหนด ACL1 และ 100 ให้กับแลนเสมือน 5 โดยให้มีการตรวจสอบกฎของ ACL1 ในทิศทางออก และการตรวจสอบกฎของ ACL100 ในทิศทางเข้า ผลลัพธ์ที่ได้คือ Config-File IX ในภาคผนวก ก มีขั้นตอนการตั้งค่าดังนี้

ขั้นที่1 เลือกเมนู Create/Remove VLAN, เลือกแลนเสมือน 5, เลือกปุ่ม Edit

ขั้นที่2 เลือกปุ่ม ACL เพื่อกำหนด ACL

ขั้นที่3 เลือก Access-List และทิศทางการตรวจสอบแพ็กเก็ต ดังรูปที่ 11-40



รูปที่ 11-41 แสดงการเลือก ACL และกำหนดทิศทางในการตรวจสอบแพ็กเก็ต

ขั้นที่4 สร้าง Configuration File และสังเกตผลที่ได้ดังรูปที่ 11-41

```
160>    interface Vlan5
161>    ip address 161.246.5.252 255.255.255.0
162>    ip access-group 1 out ....(1)
163>    ip access-group 100 in ....(2)
```

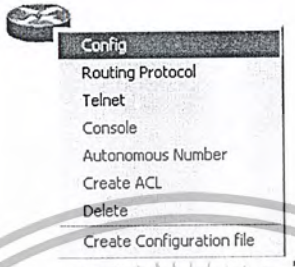
รูปที่ 11-42 แสดงค่าติดตั้งบางส่วนจากภาคผนวก ก Config-File IX

11.2. ตัวอย่างทดสอบการตั้งค่าเราเตอร์

11.2.1 ทดสอบการกำหนดรายละเอียดทั่วไปของเราเตอร์

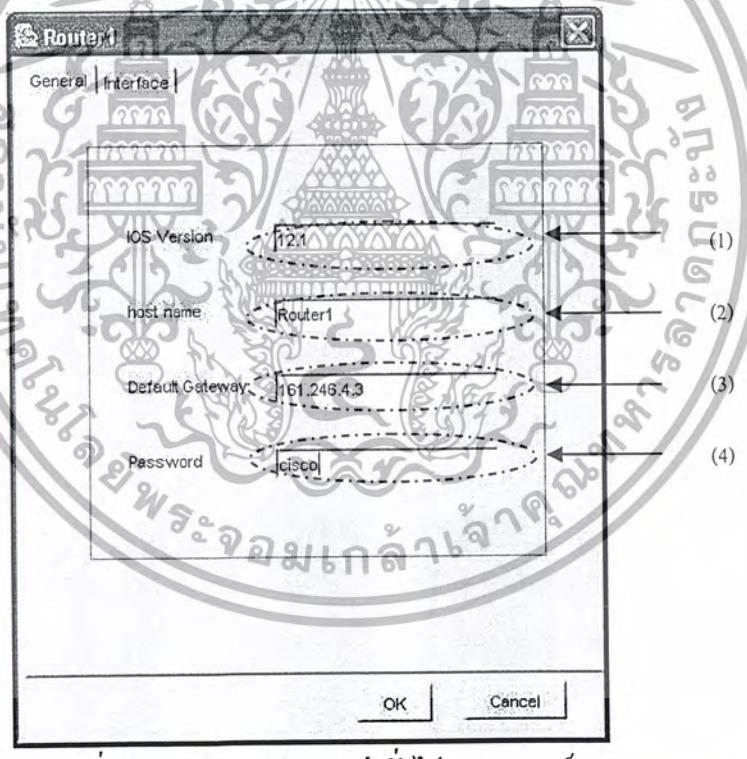
ตัวอย่างนี้เป็นการสร้างค่าติดตั้งเพื่อกำหนดค่า IOS Version, host name, default gateway และ password ให้กับเราเตอร์ ผลลัพธ์ที่ได้คือ Config-File X ในภาคผนวก ก มีขั้นตอนการตั้งค่าเราเตอร์ดังนี้

ขั้นที่1 เปิดโปรแกรม สร้างเราเตอร์ และเลือกเมนู config ดังรูปที่ 11-42



รูปที่ 11-43 แสดงขั้นตอนการใช้งานเมนู Config เพื่อกำหนดค่าให้เราเตอร์

ขั้นที่2 กำหนดค่า IOS Version, hostname,และ Password ดังรูปที่ 11-43



รูปที่ 11-44 แสดงการกำหนดค่าทั่วไปของเราเตอร์

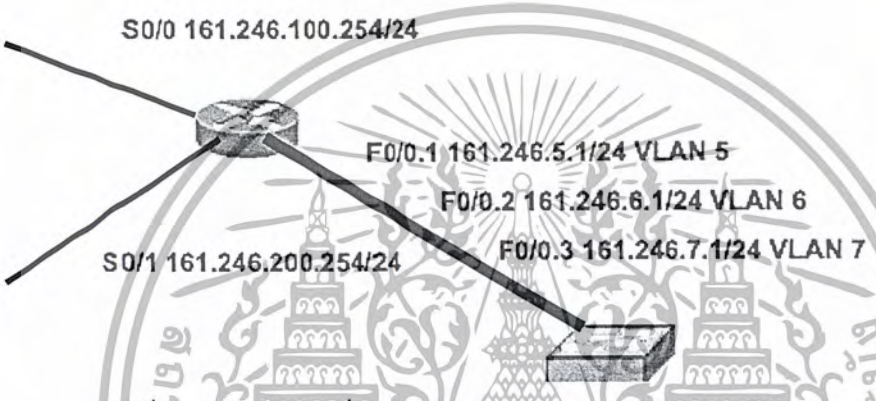
ขั้นที่3 สร้าง Configuration File และสังเกตผลที่ได้ดังรูปที่ 11-44

```
1>      version 12.1                                    .....(1)
6>      hostname Router1                                .....(2)
7>      enable password cisco                           .....(4)
        ...
29>     ip default-gateway 161.246.4.3                  .....(3)
```

รูปที่ 11-45 แสดงค่าติดตั้งบางส่วนจากภาคผนวก ก Config-File X

11.2.2 ทดสอบการกำหนดค่าบนอินเทอร์เฟสของเราเตอร์

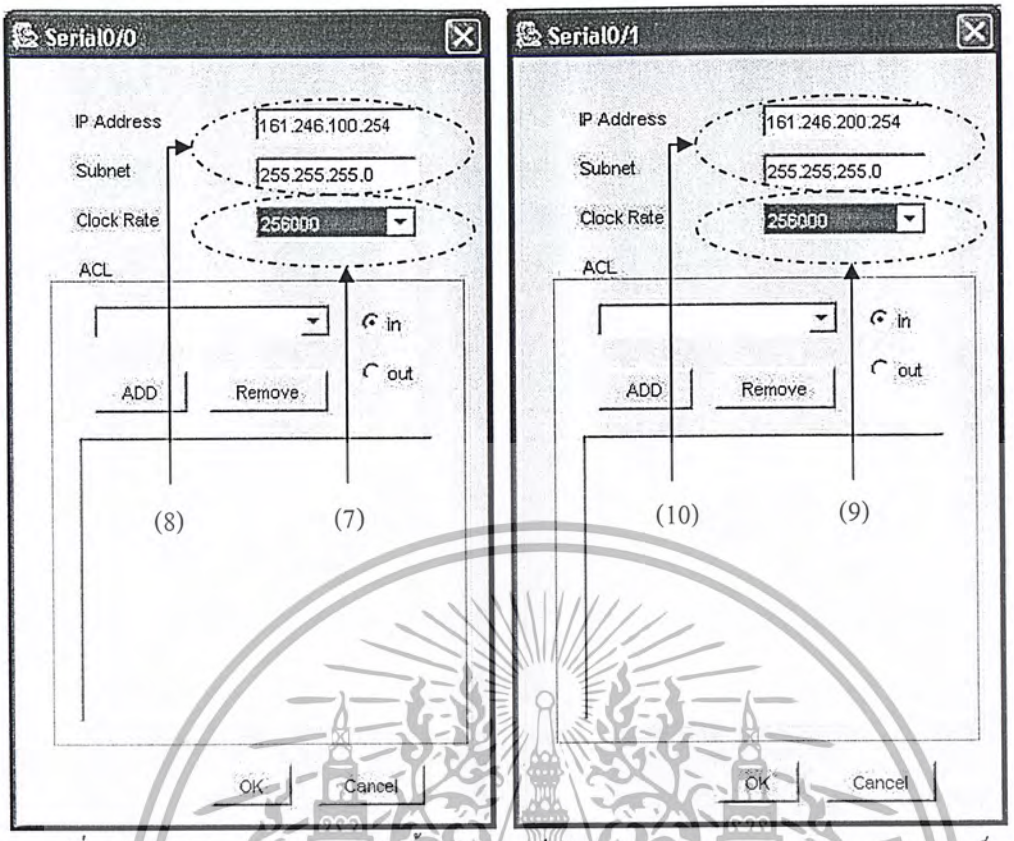
ตัวอย่างนี้เป็นการกำหนดค่าให้กับอินเทอร์เฟสต่างๆดังรูปที่ 11-45



รูปที่ 11-46 แสดงการเชื่อมต่อเครือข่ายของอินเทอร์เฟสต่างๆของเราเตอร์

Serial0/0	IP Address 161.246.100.254/24	Clock rate 256000
Serial0/1	IP Address 161.246.200.254/24	Clock rate 256000
FastEthernet0/0	เป็นทรังค์ลิงค์ ประกอบด้วยซับอินเทอร์เฟสต่างๆดังนี้	
FastEthernet0/0.1	IP Address 161.246.5.1/24	VLAN 5
FastEthernet0/0.2	IP Address 161.246.6.1/24	VLAN 6
FastEthernet0/0.2	IP Address 161.246.7.1/24	VLAN 7

<u>ขั้นที่1</u> เลือกเมนู Config เลือกแถบ Interface เลือก Serial 0/0 และเลือก Properties และกำหนดรายละเอียดดังรูปที่ 11-46

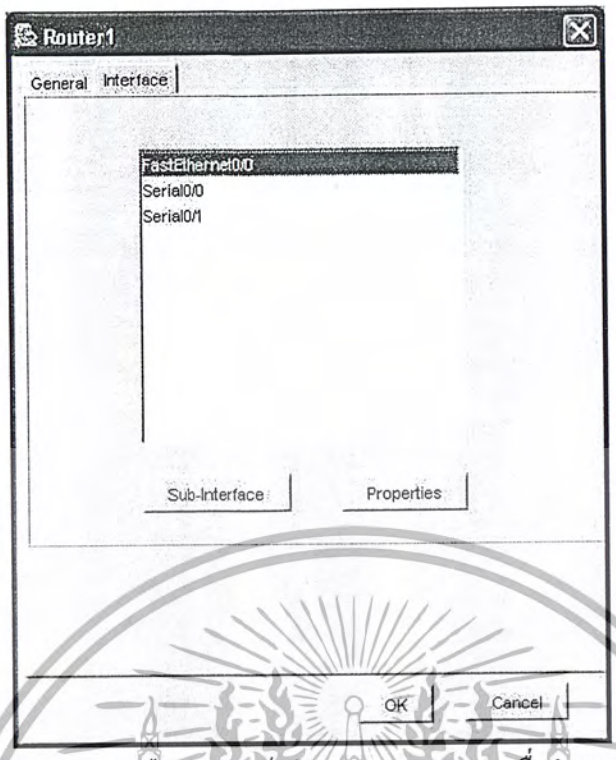


รูปที่ 11-47 แสดงการกำหนดค่าติดตั้งให้อินเทอร์เฟส Serial0/0 และ Serial0/1 ของเราเตอร์

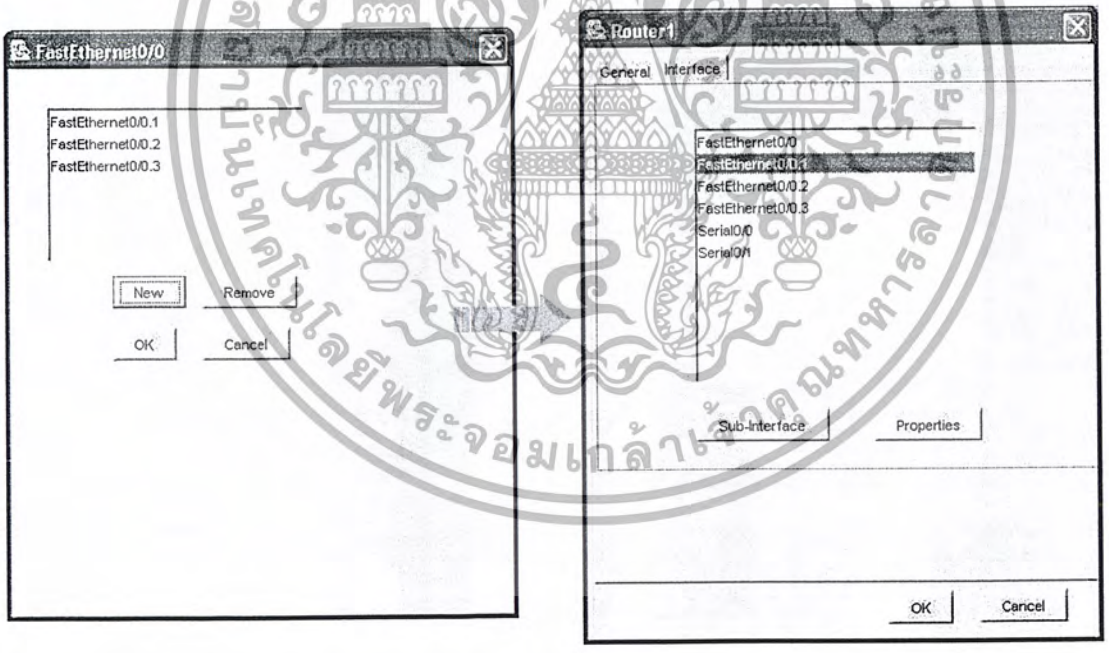
ขั้นที่2 กำหนดค่าให้กับอินเทอร์เฟส Serial 0/1 เช่นเดียวกับขั้นที่1

ขั้นที่3 สร้างซับอินเทอร์เฟสของ FastEthernet0/0 ให้กับเราเตอร์เพื่อใช้เชื่อมต่อกับทรังค์ลิงค์ที่ประกอบด้วยสมาชิกของแลนเสมือน 5, 6 และ 7 โดยเลือก FastEthernet0/0 และเลือกปุ่ม Sub-Interface ดังรูปที่ 11-47

ขั้นที่4 เลือกปุ่ม New 3 ครั้งเพื่อสร้างซับอินเทอร์เฟส 3 อินเทอร์เฟส ดังรูปที่ 11-48

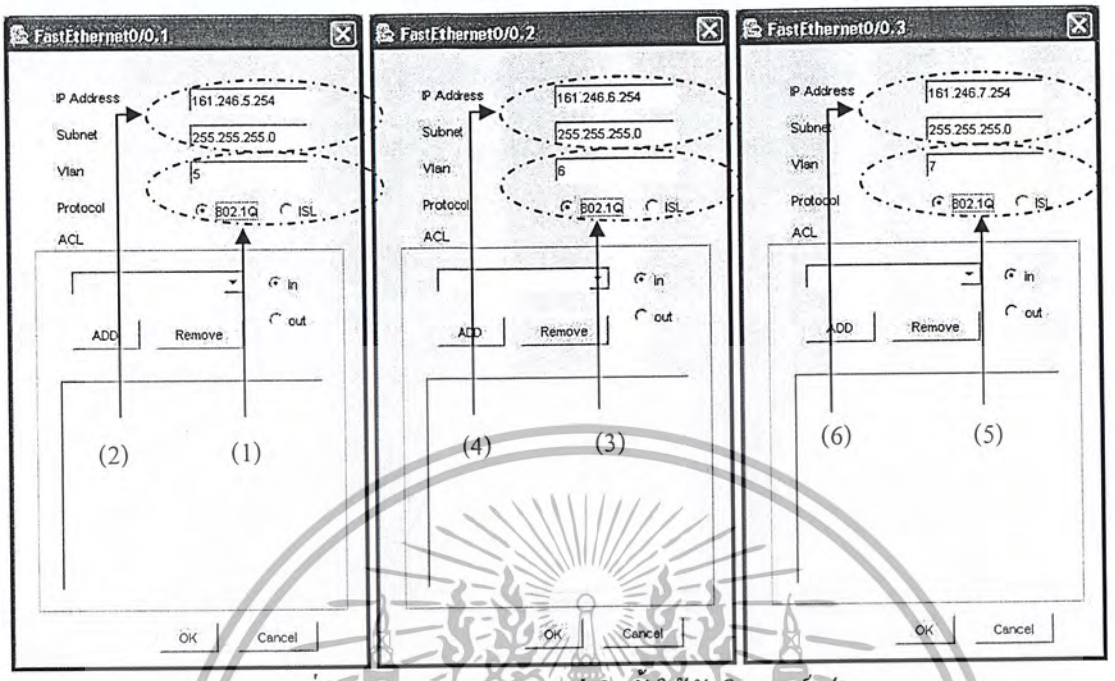


รูปที่ 11-48 แสดงการเลือกอินเทอร์เฟส FastEthernet0/0 เพื่อกำหนดค่าติดตั้ง



รูปที่ 11-49 แสดงการเลือกสร้างซับอินเทอร์เฟส FastEthernet0/0.1 , FastEthernet0/0.2 , FastEthernet0/0.3

ขั้นที่5 กำหนดรายละเอียดให้กับแต่ละซับอินเทอร์เฟส ดังรูปที่ 11-49



รูปที่ 11-50 แสดงการกำหนดค่าติดตั้งให้ซับอินเทอร์เฟส

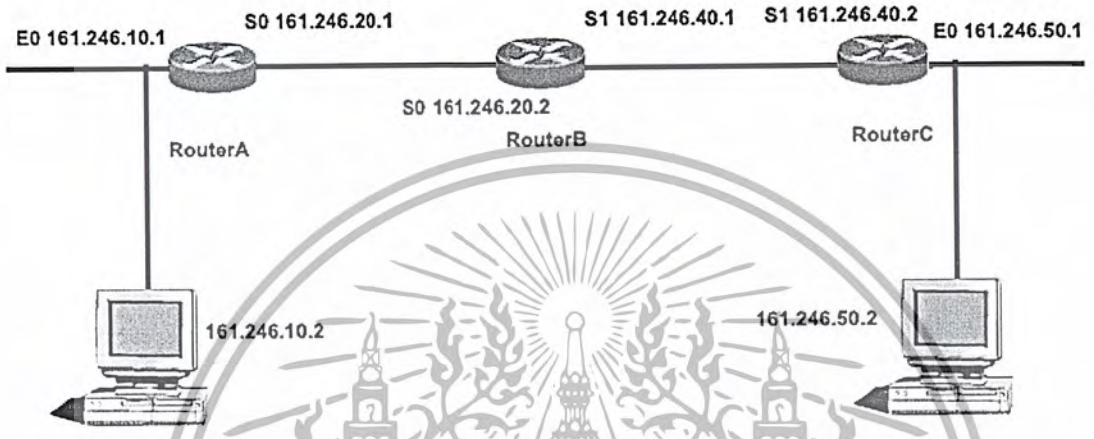
ขั้นที่6 สร้าง Configuration File และสังเกตผลที่ได้ ดังรูปที่ 11-50

```
interface FastEthernet0/0.1
encapsulation dot1Q 5
 ip address 161.246.5.254 255.255.255.0
!
interface FastEthernet0/0.2
encapsulation dot1Q 6
 ip address 161.246.6.254 255.255.255.0
!
interface FastEthernet0/0.3
encapsulation dot1Q 7
 ip address 161.246.7.254 255.255.255.0
!
interface Serial0/0
 ip address 161.246.100.254 255.255.255.0
 clockrate 256000
!
!
```

```
32>     interface Serial0/1
33>      ip address 161.246.200.254 255.255.255.0          .....(9)
34>      clockrate 256000                                  .....(10)
```

รูปที่ 11-51 แสดงค่าติดตั้งบางส่วนจากภาคผนวก ก Config-File XI

11.2.3 ทดสอบการกำหนดการเลือกเส้นทางดังรูปที่ 11-51

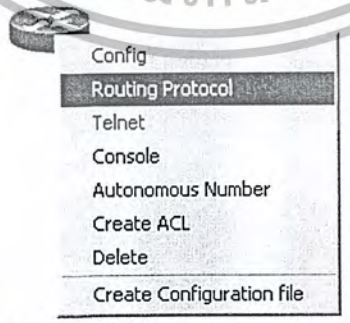


รูปที่ 11-52 เครือข่ายตัวอย่างสำหรับการกำหนดค่าเลือกเส้นทาง

11.2.3.1 ทดสอบการกำหนดดีฟอล์ตเราท์

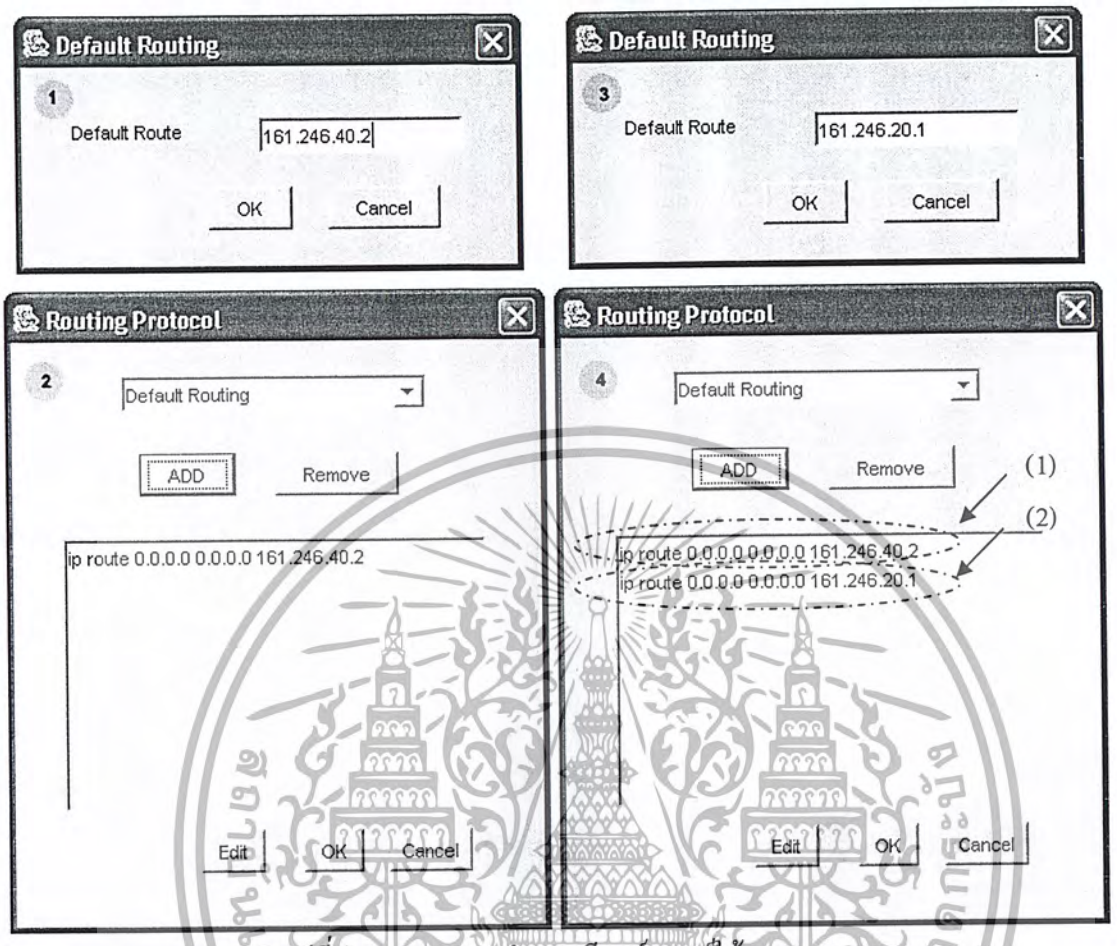
ตัวอย่างนี้เป็นการสร้างค่าติดตั้งบน RouterA เพื่อให้เราเตอร์มีการเลือกเส้นทางแบบดีฟอล์ตเราท์ โดยมีเครือข่ายเป็นดังรูปที่ 11-51 ผลลัพธ์ที่ได้คือ Config-File XII ในภาคผนวก ก มีขั้นตอนการตั้งค่าเราเตอร์ดังนี้

ขั้นที่ 1 เลือกเมนู Routing Protocol ดังรูปที่ 11-52



รูปที่ 11-53แสดงการใช้งานเมนู Routing Protocol

ขั้นที่ 2 เลือกเมนู Routing Protocol เลือกปุ่ม ADD เพื่อกำหนดค่าดังรูปที่ 11-53



รูปที่ 11-54 แสดงการกำหนดดีฟอล์ตเราท์ให้ RouterB

ขั้นที่ 3 สร้าง Configuration File และสังเกตผลที่ได้ ดังรูปที่ 11-54

```
ip route 0.0.0.0 0.0.0.0 161.246.40.2 .....(1)
ip route 0.0.0.0 0.0.0.0 161.246.20.1 .....(2)
```

รูปที่ 11-55 แสดงค่าติดตั้งบางส่วนจากภาคผนวก ก Config-File XII

ทดสอบการเลือกเส้นทางแบบสแตติก

ตัวอย่างนี้เป็นการสร้างค่าติดตั้งบนเราเตอร์ให้มีการเลือกเส้นทางแบบสแตติก โดยมีเครือข่ายเป็นดังรูปที่ 11-51 ผลลัพธ์ที่ได้คือ Config-File XIII ในภาคผนวก ก มีขั้นตอนการตั้งค่าเราเตอร์ดังนี้

ขั้นที่ 1 เลือกเมนู Routing Protocol

ขั้นที่ 2 เลือกเมนู Routing Protocol เลือกปุ่ม ADD เพื่อกำหนดค่าดังรูปที่ 11-55

รูปที่ 11-56 แสดงการกำหนดสแตติกให้ RouterB

ขั้นที่ 3 สร้าง Configuration File และสังเกตผลที่ได้ ดังรูปที่ 11-56

```
35>     ip route 161.246.50.0 255.255.255.0 161.246.40.2      .....(1)
36>     ip route 161.246.10.0 255.255.255.0 161.246.20.1      .....(2)
```

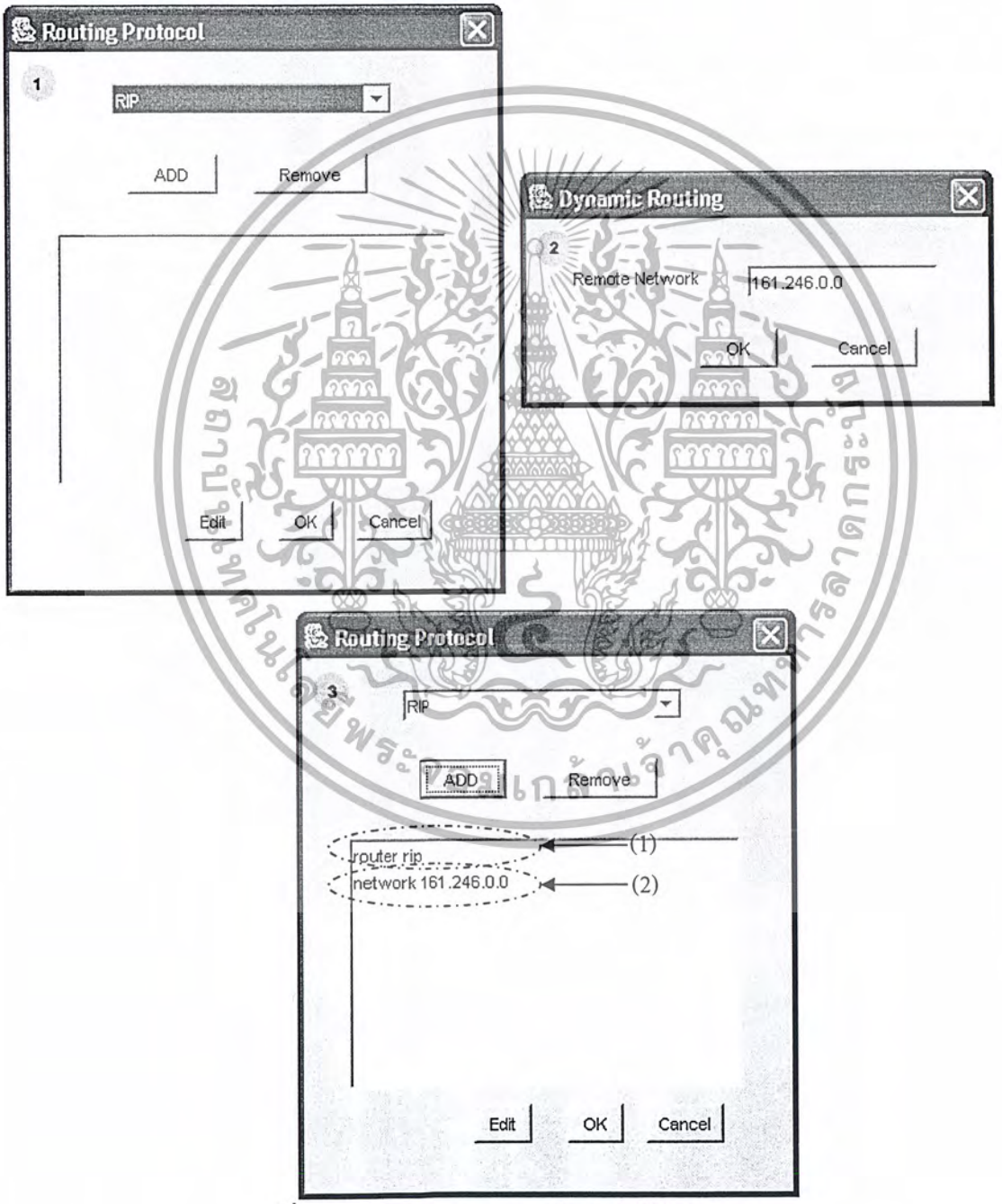
รูปที่ 11-57แสดงค่าติดตั้งบางส่วนจากภาคผนวก ก Config-File XIII

ทดสอบการเลือกเส้นทางแบบไดนามิก

ตัวอย่างนี้เป็นการสร้างค่าติดตั้งบนเราเตอร์ให้มีการเลือกเส้นทางแบบไดนามิก โดยใช้โปรโตคอลอาร์ไอพี ในเครือข่าย 161.246.0.0 ผลลัพธ์ที่ได้คือ Config-File XIV ในภาคผนวก ก มีขั้นตอนการตั้งค่าเราเตอร์ดังนี้

ขั้นที่ 1 เลือกเมนู Routing Protocol

ขั้นที่ 2 เลือกเมนู Routing Protocol เลือกปุ่ม ADD เพื่อกำหนดค่าดังรูปที่ 11-57



รูปที่ 11-58 แสดงการกำหนดอาร์ไอพีเราท์ให้ RouterB

ขั้นที่ 3 สร้าง Configuration File และสังเกตผลที่ได้ ดังรูปที่ 11-58

```
33>     router rip                                    .....(1)
34>      network 161.246.0.0                          .....(2)
```

รูปที่ 11-59 แสดงค่าติดตั้งบางส่วนจากภาคผนวก ก Config-File XIV

ภาคผนวก ก

Config-file I

```
>1    version 12.1
>2    no service single-slot-reload-enable
>3    no service pad
>4    service timestamps debug uptime
>5    service timestamps log uptime
>6    no service password-encryption
>7    !
>8    hostname Switch1
>9    enable password cisco
>10   !
>11   ip subnet-zero
>12   spanning-tree extend system-id
>13   !
>14   !
>15   interface FastEthernet0/1
>16   no ip address
>17   !
>18   interface FastEthernet0/2
>19   no ip address
>20   !
>21   interface FastEthernet0/3
>22   no ip address
>23   !
>24   interface FastEthernet0/4
>25   no ip address
>26   !
>27   interface FastEthernet0/5
>28   no ip address
>29   !
>30   interface FastEthernet0/6
>31   no ip address
>32   !
>33   interface FastEthernet0/7
```

```
>34      no ip address
>35      !
>36      interface FastEthernet0/8
>37      no ip address
>38      !
>39      interface FastEthernet0/9
>40      no ip address
>41      !
>42      interface FastEthernet0/10
>43      no ip address
>44      !
>45      interface FastEthernet0/11
>46      no ip address
>47      !
>48      interface FastEthernet0/12
>49      no ip address
>50      !
>51      interface FastEthernet0/13
>52      no ip address
>53      !
>54      interface FastEthernet0/14
>55      no ip address
>56      !
>57      interface FastEthernet0/15
>58      no ip address
>59      !
>60      interface FastEthernet0/16
>61      no ip address
>62      !
>63      interface FastEthernet0/17
>64      no ip address
>65      !
>66      interface FastEthernet0/18
>67      no ip address
>68      !
>69      interface FastEthernet0/19
>70      no ip address
```

```
>71     !
>72     interface FastEthernet0/20
>73     no ip address
>74     !
>75     interface FastEthernet0/21
>76     no ip address
>77     !
>78     interface FastEthernet0/22
>79     no ip address
>80     !
>81     interface FastEthernet0/23
>82     no ip address
>83     !
>84     interface FastEthernet0/24
>85     no ip address
>86     !
>87     interface Vlan1
>88     no ip address
>89     no ip route-cache
>90     !
>91     ip default-gateway 161.246.5.252
>92     ip http server
>93     !
>94     line con 0
>95     line vty 0 15
>96     !
>97     end
>98
>99
>100
>101
>102
>103
>104
>105
>106
>107
```

Config-file II

```
>1    version 12.1
>2    no service single-slot-reload-enable
>3    no service pad
>4    service timestamps debug uptime
>5    service timestamps log uptime
>6    no service password-encryption
>7    !
>8    hostname Switch1
>9    !
>10   enable password cisco
>11   ip subnet-zero
>12   spanning-tree extend system-id
>13   !
>14   !
>15   interface FastEthernet0/1
>16   no ip address
>17   !
>18   interface FastEthernet0/2
>19   no ip address
>20   !
>21   interface FastEthernet0/3
>22   no ip address
>23   !
>24   interface FastEthernet0/4
>25   no ip address
>26   !
>27   interface FastEthernet0/5
>28   no ip address
>29   !
>30   interface FastEthernet0/6
>31   no ip address
>32   !
>33   interface FastEthernet0/7
>34   no ip address
>35   !
```

```
>36         interface FastEthernet0/8
>37         no ip address
>38         !
>39         interface FastEthernet0/9
>40         no ip address
>41         !
>42         interface FastEthernet0/10
>43         no ip address
>44         !
>45         interface FastEthernet0/11
>46         no ip address
>47         !
>48         interface FastEthernet0/12
>49         no ip address
>50         !
>51         interface FastEthernet0/13
>52         no ip address
>53         !
>54         interface FastEthernet0/14
>55         no ip address
>56         !
>57         interface FastEthernet0/15
>58         no ip address
>59         !
>60         interface FastEthernet0/16
>61         no ip address
>62         !
>63         interface FastEthernet0/17
>64         no ip address
>65         !
>66         interface FastEthernet0/18
>67         no ip address
>68         !
>69         interface FastEthernet0/19
>70         no ip address
>71         !
>72         interface FastEthernet0/20
```

```
>73      no ip address
>74      !
>75      interface FastEthernet0/21
>76      no ip address
>77      !
>78      interface FastEthernet0/22
>79      no ip address
>80      !
>81      interface FastEthernet0/23
>82      no ip address
>83      !
>84      interface FastEthernet0/24
>85      no ip address
>86      !
>87      interface Vlan1
>88      no ip address
>89      no ip route-cache
>90      !
>91      interface Vlan5
>92      ip address 161.246.5.252 255.255.255.0
>93      no ip route-cache
>94      !
>95      interface Vlan6
>96      no ip address
>97      no ip route-cache
>98      !
>99      interface Vlan7
>100     no ip address
>101     no ip route-cache
>102     !
>103     ip default-gateway 161.246.5.252
>104     ip http server
>105     !
>106     line con 0
>107     line vty 0 15
>108     !
>109     end
```

Config-file III

```
>1    version 12.1
>2    no service single-slot-reload-enable
>3    no service pad
>4    service timestamps debug uptime
>5    service timestamps log uptime
>6    no service password-encryption
>7    !
>8    hostname Switch1
>9    enable password cisco
>10   !
>11   ip subnet-zero
>12   spanning-tree extend system-id
>13   !
>14   !
>15   interface FastEthernet0/1
>16   switchport acces vlan 5
>17   switchport mode access
>18   no ip address
>19   !
>20   interface FastEthernet0/2
>21   no ip address
>22   !
>23   interface FastEthernet0/3
>24   no ip address
>25   !
>26   interface FastEthernet0/4
>27   no ip address
>28   !
>29   interface FastEthernet0/5
>30   no ip address
>31   !
>32   interface FastEthernet0/6
>33   no ip address
>34   !
>35   interface FastEthernet0/7
```

```
>36         no ip address
>37         !
>38         interface FastEthernet0/8
>39         no ip address
>40         !
>41         interface FastEthernet0/9
>42         no ip address
>43         !
>44         interface FastEthernet0/10
>45         no ip address
>46         !
>47         interface FastEthernet0/11
>48         no ip address
>49         !
>50         interface FastEthernet0/12
>51         no ip address
>52         !
>53         interface FastEthernet0/13
>54         no ip address
>55         !
>56         interface FastEthernet0/14
>57         no ip address
>58         !
>59         interface FastEthernet0/15
>60         no ip address
>61         !
>62         interface FastEthernet0/16
>63         no ip address
>64         !
>65         interface FastEthernet0/17
>66         no ip address
>67         !
>68         interface FastEthernet0/18
>69         no ip address
>70         !
>71         interface FastEthernet0/19
>72         no ip address
```

```
>73      !
>74      interface FastEthernet0/20
>75      no ip address
>76      !
>77      interface FastEthernet0/21
>78      no ip address
>79      !
>80      interface FastEthernet0/22
>81      no ip address
>82      !
>83      interface FastEthernet0/23
>84      no ip address
>85      !
>86      interface FastEthernet0/24
>87      no ip address
>88      !
>89      interface Vlan1
>90      no ip address
>91      no ip route-cache
>92      !
>93      interface Vlan5
>94      ip address 161.246.5.252 255.255.255.0
>95      no ip route-cache
>96      !
>97      interface Vlan6
>98      no ip address
>99      no ip route-cache
>100     !
>101     interface Vlan7
>102     no ip address
>103     no ip route-cache
>104     !
>105     ip default-gateway 161.246.5.252
>106     ip http server
>107     !
>108     line con 0
>109     line vty 0 15
```

```
>110        !
>111        end
>112
>113
>114
>115
>116
>117
>118
>119
>120
>121
>122
>123
>124
>125
>126
>127
>128
>129
>130
>131
>132
>133
>134
>135
>136
>137
>138
>139
>140
>141
>142
>143
>144
>145
>146
```

Config-file IV

```
>1    version 12.1
>2    no service single-slot-reload-enable
>3    no service pad
>4    service timestamps debug uptime
>5    service timestamps log uptime
>6    no service password-encryption
>7    !
>8    hostname Switch1
>9    enable password cisco
>10   !
>11   ip subnet-zero
>12   spanning-tree extend system-id
>13   !
>14   !
>15   interface Port-channel1
>16   switchport trunk allowed vlan 5,6,7
>17   switchport mode trunk
>18   no ip address
>19   flowcontrol send off
>20   spanning-tree vlan 5 port-priority 7
>21   spanning-tree vlan 6 port-priority 7
>22   spanning-tree vlan 7 port-priority 3
>23   !
>24   interface Port-channel2
>25   switchport trunk allowed vlan 5,6,7
>26   switchport mode trunk
>27   no ip address
>28   flowcontrol send off
>29   spanning-tree vlan 5 port-priority 3
>30   spanning-tree vlan 6 port-priority 7
>31   spanning-tree vlan 7 port-priority 7
>32   !
>33   interface FastEthernet0/1
>34   switchport acces vlan 5
>35   switchport mode access
```

```
>36      no ip address
>37      !
>38      interface FastEthernet0/2
>39      no ip address
>40      !
>41      interface FastEthernet0/3
>42      no ip address
>43      !
>44      interface FastEthernet0/4
>45      no ip address
>46      !
>47      interface FastEthernet0/5
>48      no ip address
>49      !
>50      interface FastEthernet0/6
>51      no ip address
>52      !
>53      interface FastEthernet0/7
>54      switchport acces vlan 5
>55      switchport mode access
>56      no ip address
>57      !
>58      interface FastEthernet0/8
>59      switchport trunk allowed vlan 5,6,7
>60      switchport mode trunk
>61      no ip address
>62      channel-group 1 mode on
>63      spanning-tree vlan 5 port-priority 7
>64      spanning-tree vlan 6 port-priority 7
>65      spanning-tree vlan 7 port-priority 3
>66      !
>67      interface FastEthernet0/9
>68      switchport trunk allowed vlan 5,6,7
>69      switchport mode trunk
>70      no ip address
>71      channel-group 1 mode on
>72      spanning-tree vlan 5 port-priority 7
```

```
>73      spanning-tree vlan 6 port-priority 7
>74      spanning-tree vlan 7 port-priority 3
>75      !
>76      interface FastEthernet0/10
>77      no ip address
>78      !
>79      interface FastEthernet0/11
>80      no ip address
>81      !
>82      interface FastEthernet0/12
>83      no ip address
>84      !
>85      interface FastEthernet0/13
>86      no ip address
>87      !
>88      interface FastEthernet0/14
>89      no ip address
>90      !
>91      interface FastEthernet0/15
>92      no ip address
>93      !
>94      interface FastEthernet0/16
>95      switchport trunk allowed vlan 5,6,7
>96      switchport mode trunk
>97      no ip address
>98      channel-group 2 mode on
>99      spanning-tree vlan 5 port-priority 3
>100     spanning-tree vlan 6 port-priority 7
>101     spanning-tree vlan 7 port-priority 7
>102     !
>103     interface FastEthernet0/17
>104     switchport trunk allowed vlan 5,6,7
>105     switchport mode trunk
>106     no ip address
>107     channel-group 2 mode on
>108     spanning-tree vlan 5 port-priority 3
>109     spanning-tree vlan 6 port-priority 7
```

```
>110    spanning-tree vlan 7 port-priority 7
>111    !
>112    interface FastEthernet0/18
>113    no ip address
>114    !
>115    interface FastEthernet0/19
>116    no ip address
>117    !
>118    interface FastEthernet0/20
>119    no ip address
>120    !
>121    interface FastEthernet0/21
>122    no ip address
>123    !
>124    interface FastEthernet0/22
>125    no ip address
>126    !
>127    interface FastEthernet0/23
>128    no ip address
>129    !
>130    interface FastEthernet0/24
>131    no ip address
>132    !
>133    interface Vlan1
>134    no ip address
>135    no ip route-cache
>136    !
>137    interface Vlan5
>138    ip address 161.246.5.252 255.255.255.0
>139    no ip route-cache
>140    !
>141    interface Vlan6
>142    no ip address
>143    no ip route-cache
>144    !
>145    interface Vlan7
>146    no ip address
```

```
>147        no ip route-cache
>148        !
>149        ip default-gateway 161.246.5.252
>150        ip http server
>151        !
>152        line con 0
>153        line vty 0 15
>154        !
>155        end
>156
>157
>158
>159
>160
>161
>162
>163
>164
>165
>166
>167
>168
>169
>170
>171
>172
>173
>174
>175
>176
>177
>178
>179
>180
>181
>182
>183
```

Config-file V

```
>1    version 12.1
>2    no service single-slot-reload-enable
>3    no service pad
>4    service timestamps debug uptime
>5    service timestamps log uptime
>6    no service password-encryption
>7    !
>8    hostname Switch1
>9    enable password cisco
>10   !
>11   ip subnet-zero
>12   spanning-tree extend system-id
>13   !
>14   !
>15   interface Port-channel1
>16   switchport trunk allowed vlan 5,6,7
>17   switchport mode trunk
>18   no ip address
>19   flowcontrol send off
>20   spanning-tree vlan 5 port-priority 7
>21   spanning-tree vlan 6 port-priority 7
>22   spanning-tree vlan 7 port-priority 3
>23   !
>24   interface Port-channel2
>25   switchport trunk allowed vlan 5,6,7
>26   switchport mode trunk
>27   no ip address
>28   flowcontrol send off
>29   spanning-tree vlan 5 port-priority 3
>30   spanning-tree vlan 6 port-priority 7
>31   spanning-tree vlan 7 port-priority 7
>32   !
>33   interface FastEthernet0/1
>34   switchport acces vlan 5
>35   switchport mode access
```

```
>36      no ip address
>37      !
>38      interface FastEthernet0/2
>39      no ip address
>40      !
>41      interface FastEthernet0/3
>42      no ip address
>43      !
>44      interface FastEthernet0/4
>45      no ip address
>46      !
>47      interface FastEthernet0/5
>48      no ip address
>49      !
>50      interface FastEthernet0/6
>51      no ip address
>52      !
>53      interface FastEthernet0/7
>54      switchport acces vlan 5
>55      switchport mode access
>56      no ip address
>57      !
>58      interface FastEthernet0/8
>59      switchport trunk allowed vlan 5,6,7
>60      switchport mode trunk
>61      no ip address
>62      channel-group 1 mode on
>63      spanning-tree vlan 5 port-priority 7
>64      spanning-tree vlan 6 port-priority 7
>65      spanning-tree vlan 7 port-priority 3
>66      !
>67      interface FastEthernet0/9
>68      switchport trunk allowed vlan 5,6,7
>69      switchport mode trunk
>70      no ip address
>71      channel-group 1 mode on
>72      spanning-tree vlan 5 port-priority 7
```

```
>73      spanning-tree vlan 6 port-priority 7
>74      spanning-tree vlan 7 port-priority 3
>75      !
>76      interface FastEthernet0/10
>77      no ip address
>78      !
>79      interface FastEthernet0/11
>80      no ip address
>81      !
>82      interface FastEthernet0/12
>83      no ip address
>84      !
>85      interface FastEthernet0/13
>86      no ip address
>87      !
>88      interface FastEthernet0/14
>89      no ip address
>90      !
>91      interface FastEthernet0/15
>92      no ip address
>93      !
>94      interface FastEthernet0/16
>95      switchport trunk allowed vlan 5,6,7
>96      switchport mode trunk
>97      no ip address
>98      channel-group 2 mode on
>99      spanning-tree vlan 5 port-priority 3
>100     spanning-tree vlan 6 port-priority 7
>101     spanning-tree vlan 7 port-priority 7
>102     !
>103     interface FastEthernet0/17
>104     switchport trunk allowed vlan 5,6,7
>105     switchport mode trunk
>106     no ip address
>107     channel-group 2 mode on
>108     spanning-tree vlan 5 port-priority 3
>109     spanning-tree vlan 6 port-priority 7
```

```
>110    spanning-tree vlan 7 port-priority 7
>111    !
>112    interface FastEthernet0/18
>113    no ip address
>114    !
>115    interface FastEthernet0/19
>116    no ip address
>117    !
>118    interface FastEthernet0/20
>119    no ip address
>120    !
>121    interface FastEthernet0/21
>122    no ip address
>123    !
>124    interface FastEthernet0/22
>125    no ip address
>126    !
>127    interface FastEthernet0/23
>128    no ip address
>129    !
>130    interface FastEthernet0/24
>131    no ip address
>132    !
>133    interface Vlan1
>134    no ip address
>135    no ip route-cache
>136    !
>137    interface Vlan5
>138    ip address 161.246.5.252 255.255.255.0
>139    no ip route-cache
>140    !
>141    interface Vlan6
>142    no ip address
>143    no ip route-cache
>144    !
>145    interface Vlan7
>146    no ip address
```

```
>147        no ip route-cache
>148        !
>149        ip default-gateway 161.246.5.252
>150        ip http server
>151        !
>152        line con 0
>153        line vty 0 15
>154        password cisco
>155        login
>156        !
>157        end
>158
>159
>160
>161
>162
>163
>164
>165
>166
>167
>168
>169
>170
>171
>172
>173
>174
>175
>176
>177
>178
>179
>180
>181
>182
>183
```

Config-file VI

```
>1    version 12.1
>2    no service single-slot-reload-enable
>3    no service pad
>4    service timestamps debug uptime
>5    service timestamps log uptime
>6    no service password-encryption
>7    !
>8    hostname Switch1
>9    enable password cisco
>10   !
>11   ip subnet-zero
>12   spanning-tree extend system-id
>13   !
>14   !
>15   interface Port-channel1
>16   switchport trunk allowed vlan 5,6,7
>17   switchport mode trunk
>18   no ip address
>19   flowcontrol send off
>20   spanning-tree vlan 5 port-priority 7
>21   spanning-tree vlan 6 port-priority 7
>22   spanning-tree vlan 7 port-priority 3
>23   !
>24   interface Port-channel2
>25   switchport trunk allowed vlan 5,6,7
>26   switchport mode trunk
>27   no ip address
>28   flowcontrol send off
>29   spanning-tree vlan 5 port-priority 3
>30   spanning-tree vlan 6 port-priority 7
>31   spanning-tree vlan 7 port-priority 7
>32   !
>33   interface FastEthernet0/1
>34   switchport acces vlan 5
>35   switchport mode access
```

```
>36      no ip address
>37      !
>38      interface FastEthernet0/2
>39      no ip address
>40      !
>41      interface FastEthernet0/3
>42      no ip address
>43      !
>44      interface FastEthernet0/4
>45      no ip address
>46      !
>47      interface FastEthernet0/5
>48      no ip address
>49      !
>50      interface FastEthernet0/6
>51      no ip address
>52      !
>53      interface FastEthernet0/7
>54      switchport acces vlan 5
>55      switchport mode access
>56      no ip address
>57      !
>58      interface FastEthernet0/8
>59      switchport trunk allowed vlan 5,6,7
>60      switchport mode trunk
>61      no ip address
>62      channel-group 1 mode on
>63      spanning-tree vlan 5 port-priority 7
>64      spanning-tree vlan 6 port-priority 7
>65      spanning-tree vlan 7 port-priority 3
>66      !
>67      interface FastEthernet0/9
>68      switchport trunk allowed vlan 5,6,7
>69      switchport mode trunk
>70      no ip address
>71      channel-group 1 mode on
>72      spanning-tree vlan 5 port-priority 7
```

```
>73      spanning-tree vlan 6 port-priority 7
>74      spanning-tree vlan 7 port-priority 3
>75      !
>76      interface FastEthernet0/10
>77      no ip address
>78      !
>79      interface FastEthernet0/11
>80      no ip address
>81      !
>82      interface FastEthernet0/12
>83      no ip address
>84      !
>85      interface FastEthernet0/13
>86      no ip address
>87      !
>88      interface FastEthernet0/14
>89      no ip address
>90      !
>91      interface FastEthernet0/15
>92      no ip address
>93      !
>94      interface FastEthernet0/16
>95      switchport trunk allowed vlan 5,6,7
>96      switchport mode trunk
>97      no ip address
>98      channel-group 2 mode on
>99      spanning-tree vlan 5 port-priority 3
>100     spanning-tree vlan 6 port-priority 7
>101     spanning-tree vlan 7 port-priority 7
>102     !
>103     interface FastEthernet0/17
>104     switchport trunk allowed vlan 5,6,7
>105     switchport mode trunk
>106     no ip address
>107     channel-group 2 mode on
>108     spanning-tree vlan 5 port-priority 3
>109     spanning-tree vlan 6 port-priority 7
```

```
>110    spanning-tree vlan 7 port-priority 7
>111    !
>112    interface FastEthernet0/18
>113    no ip address
>114    !
>115    interface FastEthernet0/19
>116    no ip address
>117    !
>118    interface FastEthernet0/20
>119    no ip address
>120    !
>121    interface FastEthernet0/21
>122    no ip address
>123    !
>124    interface FastEthernet0/22
>125    no ip address
>126    !
>127    interface FastEthernet0/23
>128    no ip address
>129    !
>130    interface FastEthernet0/24
>131    no ip address
>132    !
>133    interface Vlan1
>134    no ip address
>135    no ip route-cache
>136    !
>137    interface Vlan5
>138    ip address 161.246.5.252 255.255.255.0
>139    no ip route-cache
>140    !
>141    interface Vlan6
>142    no ip address
>143    no ip route-cache
>144    !
>145    interface Vlan7
>146    no ip address
```

```
>147    no ip route-cache
>148    !
>149    ip default-gateway 161.246.5.252
>150    ip http server
>151    !
>152    line con 0
>153    password abc123
>154    login
>155    line vty 0 15
>156    password cisco
>157    login
>158    !
>159    end
>160
>161
>162
>163
>164
>165
>166
>167
>168
>169
>170
>171
>172
>173
>174
>175
>176
>177
>178
>179
>180
>181
>182
>183
```

Config-file VII

```
>1    version 12.1
>2    no service single-slot-reload-enable
>3    no service pad
>4    service timestamps debug uptime
>5    service timestamps log uptime
>6    no service password-encryption
>7    !
>8    hostname Switch1
>9    enable password cisco
>10   !
>11   ip subnet-zero
>12   spanning-tree extend system-id
>13   !
>14   !
>15   interface Port-channel1
>16   switchport trunk allowed vlan 5,6,7
>17   switchport mode trunk
>18   no ip address
>19   flowcontrol send off
>20   spanning-tree vlan 5 port-priority 7
>21   spanning-tree vlan 6 port-priority 7
>22   spanning-tree vlan 7 port-priority 3
>23   !
>24   interface Port-channel2
>25   switchport trunk allowed vlan 5,6,7
>26   switchport mode trunk
>27   no ip address
>28   flowcontrol send off
>29   spanning-tree vlan 5 port-priority 3
>30   spanning-tree vlan 6 port-priority 7
>31   spanning-tree vlan 7 port-priority 7
>32   !
>33   interface FastEthernet0/1
>34   switchport acces vlan 5
>35   switchport mode access
```

```
>36     no ip address
>37     !
>38     interface FastEthernet0/2
>39     switchport acces vlan 5
>40     switchport mode access
>41     no ip address
>42     !
>43     interface FastEthernet0/3
>44     switchport acces vlan 5
>45     switchport mode access
>46     no ip address
>47     !
>48     interface FastEthernet0/4
>49     switchport acces vlan 5
>50     switchport mode access
>51     no ip address
>52     !
>53     interface FastEthernet0/5
>54     switchport acces vlan 5
>55     switchport mode access
>56     no ip address
>57     !
>58     interface FastEthernet0/6
>59     switchport acces vlan 5
>60     switchport mode access
>61     no ip address
>62     !
>63     interface FastEthernet0/7
>64     switchport acces vlan 5
>65     switchport mode access
>66     no ip address
>67     !
>68     interface FastEthernet0/8
>69     switchport trunk allowed vlan 5,6,7
>70     switchport mode trunk
>71     no ip address
>72     channel-group 1 mode on
```

```
>73      spanning-tree vlan 5 port-priority 7
>74      spanning-tree vlan 6 port-priority 7
>75      spanning-tree vlan 7 port-priority 3
>76      !
>77      interface FastEthernet0/9
>78      switchport trunk allowed vlan 5,6,7
>79      switchport mode trunk
>80      no ip address
>81      channel-group 1 mode on
>82      spanning-tree vlan 5 port-priority 7
>83      spanning-tree vlan 6 port-priority 7
>84      spanning-tree vlan 7 port-priority 3
>85      !
>86      interface FastEthernet0/10
>87      switchport acces vlan 6
>88      switchport mode access
>89      no ip address
>90      !
>91      interface FastEthernet0/11
>92      switchport acces vlan 6
>93      switchport mode access
>94      no ip address
>95      !
>96      interface FastEthernet0/12
>97      switchport acces vlan 6
>98      switchport mode access
>99      no ip address
>100     !
>101     interface FastEthernet0/13
>102     switchport acces vlan 6
>103     switchport mode access
>104     no ip address
>105     !
>106     interface FastEthernet0/14
>107     switchport acces vlan 6
>108     switchport mode access
>109     no ip address
```

```
>110    !
>111    interface FastEthernet0/15
>112    switchport acces vlan 6
>113    switchport mode access
>114    no ip address
>115    !
>116    interface FastEthernet0/16
>117    switchport trunk allowed vlan 5,6,7
>118    switchport mode trunk
>119    no ip address
>120    channel-group 2 mode on
>121    spanning-tree vlan 5 port-priority 3
>122    spanning-tree vlan 6 port-priority 7
>123    spanning-tree vlan 7 port-priority 7
>124    !
>125    interface FastEthernet0/17
>126    switchport trunk allowed vlan 5,6,7
>127    switchport mode trunk
>128    no ip address
>129    channel-group 2 mode on
>130    spanning-tree vlan 5 port-priority 3
>131    spanning-tree vlan 6 port-priority 7
>132    spanning-tree vlan 7 port-priority 7
>133    !
>134    interface FastEthernet0/18
>135    no ip address
>136    !
>137    interface FastEthernet0/19
>138    no ip address
>139    !
>140    interface FastEthernet0/20
>141    no ip address
>142    !
>143    interface FastEthernet0/21
>144    no ip address
>145    !
>146    interface FastEthernet0/22
```

```
>147    no ip address
>148    !
>149    interface FastEthernet0/23
>150    no ip address
>151    !
>152    interface FastEthernet0/24
>153    no ip address
>154    !
>155    interface Vlan1
>156    no ip address
>157    no ip route-cache
>158    !
>159    interface Vlan5
>160    ip address 161.246.5.252 255.255.255.0
>161    no ip route-cache
>162    !
>163    interface Vlan6
>164    no ip address
>165    no ip route-cache
>166    !
>167    interface Vlan7
>168    no ip address
>169    no ip route-cache
>170    !
>171    access-list 1 permit 161.246.5.0
>172    access-list 1 permit any
>173    !
>174    ip default-gateway 161.246.5.252
>175    ip http server
>176    !
>177    line con 0
>178    password abc123
>179    login
>180    line vty 0 15
>181    password cisco
>182    login
>183    !
```

```
>184        end
>185
>186
>187
>188
>189
>190
>191
>192
>193
>194
>195
>196
>197
>198
>199
>200
>201
>202
>203
>204
>205
>206
>207
>208
>209
>210
>211
>212
>213
>214
>215
>216
>217
>218
>219
>220
```

Config-file VIII

```
>1    version 12.1
>2    no service single-slot-reload-enable
>3    no service pad
>4    service timestamps debug uptime
>5    service timestamps log uptime
>6    no service password-encryption
>7    !
>8    hostname Switch1
>9    enable password cisco
>10   !
>11   ip subnet-zero
>12   spanning-tree extend system-id
>13   !
>14   !
>15   interface Port-channel1
>16   switchport trunk allowed vlan 5,6,7
>17   switchport mode trunk
>18   no ip address
>19   flowcontrol send off
>20   spanning-tree vlan 5 port-priority 7
>21   spanning-tree vlan 6 port-priority 7
>22   spanning-tree vlan 7 port-priority 3
>23   !
>24   interface Port-channel2
>25   switchport trunk allowed vlan 5,6,7
>26   switchport mode trunk
>27   no ip address
>28   flowcontrol send off
>29   spanning-tree vlan 5 port-priority 3
>30   spanning-tree vlan 6 port-priority 7
>31   spanning-tree vlan 7 port-priority 7
>32   !
>33   interface FastEthernet0/1
>34   switchport acces vlan 5
>35   switchport mode access
```

```
>36     no ip address
>37     !
>38     interface FastEthernet0/2
>39     switchport acces vlan 5
>40     switchport mode access
>41     no ip address
>42     !
>43     interface FastEthernet0/3
>44     switchport acces vlan 5
>45     switchport mode access
>46     no ip address
>47     !
>48     interface FastEthernet0/4
>49     switchport acces vlan 5
>50     switchport mode access
>51     no ip address
>52     !
>53     interface FastEthernet0/5
>54     switchport acces vlan 5
>55     switchport mode access
>56     no ip address
>57     !
>58     interface FastEthernet0/6
>59     switchport acces vlan 5
>60     switchport mode access
>61     no ip address
>62     !
>63     interface FastEthernet0/7
>64     switchport acces vlan 5
>65     switchport mode access
>66     no ip address
>67     !
>68     interface FastEthernet0/8
>69     switchport trunk allowed vlan 5,6,7
>70     switchport mode trunk
>71     no ip address
>72     channel-group 1 mode on
```

```
>73      spanning-tree vlan 5 port-priority 7
>74      spanning-tree vlan 6 port-priority 7
>75      spanning-tree vlan 7 port-priority 3
>76      !
>77      interface FastEthernet0/9
>78      switchport trunk allowed vlan 5,6,7
>79      switchport mode trunk
>80      no ip address
>81      channel-group 1 mode on
>82      spanning-tree vlan 5 port-priority 7
>83      spanning-tree vlan 6 port-priority 7
>84      spanning-tree vlan 7 port-priority 3
>85      !
>86      interface FastEthernet0/10
>87      switchport acces vlan 6
>88      switchport mode access
>89      no ip address
>90      !
>91      interface FastEthernet0/11
>92      switchport acces vlan 6
>93      switchport mode access
>94      no ip address
>95      !
>96      interface FastEthernet0/12
>97      switchport acces vlan 6
>98      switchport mode access
>99      no ip address
>100     !
>101     interface FastEthernet0/13
>102     switchport acces vlan 6
>103     switchport mode access
>104     no ip address
>105     !
>106     interface FastEthernet0/14
>107     switchport acces vlan 6
>108     switchport mode access
>109     no ip address
```

```
>110    !
>111    interface FastEthernet0/15
>112    switchport acces vlan 6
>113    switchport mode access
>114    no ip address
>115    !
>116    interface FastEthernet0/16
>117    switchport trunk allowed vlan 5,6,7
>118    switchport mode trunk
>119    no ip address
>120    channel-group 2 mode on
>121    spanning-tree vlan 5 port-priority 3
>122    spanning-tree vlan 6 port-priority 7
>123    spanning-tree vlan 7 port-priority 7
>124    !
>125    interface FastEthernet0/17
>126    switchport trunk allowed vlan 5,6,7
>127    switchport mode trunk
>128    no ip address
>129    channel-group 2 mode on
>130    spanning-tree vlan 5 port-priority 3
>131    spanning-tree vlan 6 port-priority 7
>132    spanning-tree vlan 7 port-priority 7
>133    !
>134    interface FastEthernet0/18
>135    no ip address
>136    !
>137    interface FastEthernet0/19
>138    no ip address
>139    !
>140    interface FastEthernet0/20
>141    no ip address
>142    !
>143    interface FastEthernet0/21
>144    no ip address
>145    !
>146    interface FastEthernet0/22
```

```
>147    no ip address
>148    !
>149    interface FastEthernet0/23
>150    no ip address
>151    !
>152    interface FastEthernet0/24
>153    no ip address
>154    !
>155    interface Vlan1
>156    no ip address
>157    no ip route-cache
>158    !
>159    interface Vlan5
>160    ip address 161.246.5.252 255.255.255.0
>161    no ip route-cache
>162    !
>163    interface Vlan6
>164    no ip address
>165    no ip route-cache
>166    !
>167    interface Vlan7
>168    no ip address
>169    no ip route-cache
>170    !
>171    access-list 1 permit 161.246.5.0
>172    access-list 1 permit any
>173    access-list 100 deny TCP 161.246.50.0 0.0.0.255 any
>174    access-list 100 permit TCP any any
>175    !
>176    ip default-gateway 161.246.5.252
>177    ip http server
>178    !
>179    line con 0
>180    password abc123
>181    login
>182    line vty 0 15
>183    password cisco
```

```
>184    login
>185    !
>186    end
>187
>188
>189
>190
>191
>192
>193
>194
>195
>196
>197
>198
>199
>200
>201
>202
>203
>204
>205
>206
>207
>208
>209
>210
>211
>212
>213
>214
>215
>216
>217
>218
>219
>220
```

Config-file IX

```
>1    version 12.1
>2    no service single-slot-reload-enable
>3    no service pad
>4    service timestamps debug uptime
>5    service timestamps log uptime
>6    no service password-encryption
>7    !
>8    hostname Switch1
>9    enable password cisco
>10   !
>11   ip subnet-zero
>12   spanning-tree extend system-id
>13   !
>14   !
>15   interface Port-channel1
>16   switchport trunk allowed vlan 5,6,7
>17   switchport mode trunk
>18   no ip address
>19   flowcontrol send off
>20   spanning-tree vlan 5 port-priority 7
>21   spanning-tree vlan 6 port-priority 7
>22   spanning-tree vlan 7 port-priority 3
>23   !
>24   interface Port-channel2
>25   switchport trunk allowed vlan 5,6,7
>26   switchport mode trunk
>27   no ip address
>28   flowcontrol send off
>29   spanning-tree vlan 5 port-priority 3
>30   spanning-tree vlan 6 port-priority 7
>31   spanning-tree vlan 7 port-priority 7
>32   !
>33   interface FastEthernet0/1
>34   switchport acces vlan 5
>35   switchport mode access
```

```
>36      no ip address
>37      !
>38      interface FastEthernet0/2
>39      switchport acces vlan 5
>40      switchport mode access
>41      no ip address
>42      !
>43      interface FastEthernet0/3
>44      switchport acces vlan 5
>45      switchport mode access
>46      no ip address
>47      !
>48      interface FastEthernet0/4
>49      switchport acces vlan 5
>50      switchport mode access
>51      no ip address
>52      !
>53      interface FastEthernet0/5
>54      switchport acces vlan 5
>55      switchport mode access
>56      no ip address
>57      !
>58      interface FastEthernet0/6
>59      switchport acces vlan 5
>60      switchport mode access
>61      no ip address
>62      !
>63      interface FastEthernet0/7
>64      switchport acces vlan 5
>65      switchport mode access
>66      no ip address
>67      !
>68      interface FastEthernet0/8
>69      switchport trunk allowed vlan 5,6,7
>70      switchport mode trunk
>71      no ip address
>72      channel-group 1 mode on
```

```
>73      spanning-tree vlan 5 port-priority 7
>74      spanning-tree vlan 6 port-priority 7
>75      spanning-tree vlan 7 port-priority 3
>76      !
>77      interface FastEthernet0/9
>78      switchport trunk allowed vlan 5,6,7
>79      switchport mode trunk
>80      no ip address
>81      channel-group 1 mode on
>82      spanning-tree vlan 5 port-priority 7
>83      spanning-tree vlan 6 port-priority 7
>84      spanning-tree vlan 7 port-priority 3
>85      !
>86      interface FastEthernet0/10
>87      switchport acces vlan 6
>88      switchport mode access
>89      no ip address
>90      !
>91      interface FastEthernet0/11
>92      switchport acces vlan 6
>93      switchport mode access
>94      no ip address
>95      !
>96      interface FastEthernet0/12
>97      switchport acces vlan 6
>98      switchport mode access
>99      no ip address
>100     !
>101     interface FastEthernet0/13
>102     switchport acces vlan 6
>103     switchport mode access
>104     no ip address
>105     !
>106     interface FastEthernet0/14
>107     switchport acces vlan 6
>108     switchport mode access
>109     no ip address
```

```
>110    !
>111    interface FastEthernet0/15
>112    switchport acces vlan 6
>113    switchport mode access
>114    no ip address
>115    !
>116    interface FastEthernet0/16
>117    switchport trunk allowed vlan 5,6,7
>118    switchport mode trunk
>119    no ip address
>120    channel-group 2 mode on
>121    spanning-tree vlan 5 port-priority 3
>122    spanning-tree vlan 6 port-priority 7
>123    spanning-tree vlan 7 port-priority 7
>124    !
>125    interface FastEthernet0/17
>126    switchport trunk allowed vlan 5,6,7
>127    switchport mode trunk
>128    no ip address
>129    channel-group 2 mode on
>130    spanning-tree vlan 5 port-priority 3
>131    spanning-tree vlan 6 port-priority 7
>132    spanning-tree vlan 7 port-priority 7
>133    !
>134    interface FastEthernet0/18
>135    no ip address
>136    !
>137    interface FastEthernet0/19
>138    no ip address
>139    !
>140    interface FastEthernet0/20
>141    no ip address
>142    !
>143    interface FastEthernet0/21
>144    no ip address
>145    !
>146    interface FastEthernet0/22
```

```
>147    no ip address
>148    !
>149    interface FastEthernet0/23
>150    no ip address
>151    !
>152    interface FastEthernet0/24
>153    no ip address
>154    !
>155    interface Vlan1
>156    no ip address
>157    no ip route-cache
>158    !
>159    interface Vlan5
>160    ip address 161.246.5.252 255.255.255.0
>161    ip access-group 1 out
>162    ip access-group 100 in
>163    no ip route-cache
>164    !
>165    interface Vlan6
>166    no ip address
>167    no ip route-cache
>168    !
>169    interface Vlan7
>170    no ip address
>171    no ip route-cache
>172    !
>173    access-list 1 permit 161.246.5.0
>174    access-list 1 permit any
>175    access-list 100 deny TCP 161.246.50.0 0.0.0.255 any
>176    access-list 100 permit TCP any any
>177    ip default-gateway 161.246.5.252
>178    ip http server
>179    !
>180    line con 0
>181    password abc123
>182    login
>183    line vty 0 15
```

```
>184        password cisco
>185        login
>186        !
>187        end
>188
>189
>190
>191
>192
>193
>194
>195
>196
>197
>198
>199
>200
>201
>202
>203
>204
>205
>206
>207
>208
>209
>210
>211
>212
>213
>214
>215
>216
>217
>218
>219
>220
```

Config-file X

```
>1    version 12.1
>2    service timestamps debug datetime msec
>3    service timestamps log datetime msec
>4    no service password-encryption
>5    !
>6    hostname Router1
>7    enable password cisco
>8    !
>9    ip subnet-zero
>10   !
>11   interface FastEthernet0/0
>12    no ip address
>13    speed auto
>14   !
>15   interface Serial0/0
>16    no ip address
>17   !
>18   interface Serial0/1
>19    no ip address
>20   !
>21   interface BRI1/0
>22    no ip address
>23    shutdown
>24   !
>25   !
>26   ip classless
>27   !
>28   no ip http server
>29   ip default-gateway 161.246.4.3
>30   !
>31   line con 0
>32   line aux 0
>33   line vty 0 4
>34   !
>35   no scheduler allocate
```

```
>36     end
>37
>38
>39
>40
>41
>42
>43
>44
>45
>46
>47
>48
>49
>50
>51
>52
>53
>54
>55
>56
>57
>58
>59
>60
>61
>62
>63
>64
>65
>66
>67
>68
>69
>70
>71
>72
```

Config-file XI

```
>1    version 12.1
>2    service timestamps debug datetime msec
>3    service timestamps log datetime msec
>4    no service password-encryption
>5    !
>6    hostname Router1
>7    enable password cisco
>8    !
>9    ip subnet-zero
>10   !
>11   interface FastEthernet0/0
>12    no ip address
>13    speed auto
>14   !
>15   interface FastEthernet0/0.1
>16   encapsulation dot1Q 5
>17    ip address 161.246.5.254 255.255.255.0
>18   !
>19   interface FastEthernet0/0.2
>20   encapsulation dot1Q 6
>21    ip address 161.246.6.254 255.255.255.0
>22   !
>23   interface FastEthernet0/0.3
>24   encapsulation dot1Q 7
>25    ip address 161.246.7.254 255.255.255.0
>26   !
>27   interface Serial0/0
>28    ip address 161.246.100.254 255.255.255.0
>29    clockrate 256000
>30   !
>31   !
>32   interface Serial0/1
>33    ip address 161.246.200.254 255.255.255.0
>34    clockrate 256000
>35   !
```

```
!
interface BRI1/0
 no ip address
 shutdown
!
!
ip classless
!
no ip http server
ip default-gateway 161.246.4.3
!
line con 0
line aux 0
line vty 0 4
!
no scheduler allocate
end
```

Config-file XII

```
>1    version 12.1
>2    service timestamps debug datetime msec
>3    service timestamps log datetime msec
>4    no service password-encryption
>5    !
>6    hostname RouterB
>7    enable password cisco
>8    !
>9    ip subnet-zero
>10   !
>11   interface FastEthernet0/0
>12    no ip address
>13    speed auto
>14   !
>15   interface FastEthernet0/1
>16    no ip address
>17    speed auto
>18   !
>19   interface Serial0/0
>20    ip address 161.246.20.2 255.255.255.0
>21    clockrate 256000
>22   !
>23   !
>24   interface Serial0/1
>25    ip address 161.246.40.1 255.255.255.0
>26    clockrate 256000
>27   !
>28   !
>29   interface BRI1/0
>30    no ip address
>31    shutdown
>32   !
>33   !
>34   ip classless
>35   ip route 0.0.0.0 0.0.0.0 161.246.40.2
```

```
>36      ip route 0.0.0.0 0.0.0.0 161.246.20.1
>37      !
>38      no ip http server
>39      !
>40      line con 0
>41      line aux 0
>42      line vty 0 4
>43      !
>44      no scheduler allocate
>45      end
>46
>47
>48
>49
>50
>51
>52
>53
>54
>55
>56
>57
>58
>59
>60
>61
>62
>63
>64
>65
>66
>67
>68
>69
>70
>71
>72
```

Config-file XIII

```
>1    version 12.1
>2    service timestamps debug datetime msec
>3    service timestamps log datetime msec
>4    no service password-encryption
>5    !
>6    hostname RouterB
>7    enable password cisco
>8    !
>9    ip subnet-zero
>10   !
>11   interface FastEthernet0/0
>12    no ip address
>13    speed auto
>14   !
>15   interface FastEthernet0/1
>16    no ip address
>17    speed auto
>18   !
>19   interface Serial0/0
>20    ip address 161.246.20.2 255.255.255.0
>21    clockrate 256000
>22   !
>23   !
>24   interface Serial0/1
>25    ip address 161.246.40.1 255.255.255.0
>26    clockrate 256000
>27   !
>28   !
>29   interface BRI1/0
>30    no ip address
>31    shutdown
>32   !
>33   !
>34   ip classless
>35   ip route 161.246.50.0 255.255.255.0 161.246.40.2
```

```
ip route 161.246.10.0 255.255.255.0 161.246.20.1
!
no ip http server
!
line con 0
line aux 0
line vty 0 4
!
no scheduler allocate
end
```

Config-file XIV

```
>1      >35
>2      version 12.1
>3      service timestamps debug datetime msec
>4      service timestamps log datetime msec
>5      no service password-encryption
>6      !
>7      hostname RouterB
>8      enable password cisco
>9      !
>10     ip subnet-zero
>11     !
>12     interface FastEthernet0/0
>13      no ip address
>14      speed auto
>15     !
>16     interface FastEthernet0/1
>17      no ip address
>18      speed auto
>19     !
>20     interface Serial0/0
>21      ip address 161.246.20.2 255.255.255.0
>22      clockrate 256000
>23     !
>24     !
>25     interface Serial0/1
>26      ip address 161.246.40.1 255.255.255.0
>27      clockrate 256000
>28     !
>29     !
>30     interface BRI1/0
>31      no ip address
>32      shutdown
>33     !
>34     router rip
```

```
 network     >69
161.246.     >70
0.0          >71
!            >72
>36          ip classless
>37          !
>38          no ip http server
>39          !
>40          line con 0
>41          line aux 0
>42          line vty 0 4
>43          !
>44          no scheduler allocate
>45          end
>46
>47
>48
>49
>50
>51
>52
>53
>54
>55
>56
>57
>58
>59
>60
>61
>62
>63
>64
>65
>66
>67
>68
```

บรรณานุกรม

หนังสืออ้างอิง

[1] วีระศักดิ์ ซึงถาวร,"Java Programming Volume 1", ซีเอ็ดยูเคชั่น,432 หน้า.2543.

[2] วีระศักดิ์ ซึงถาวร,"Java Programming Volume 2", ซีเอ็ดยูเคชั่น,432 หน้า.2545.

[3] สุรศักดิ์ สงวนพงษ์,"สถาปัตยกรรมและโพรโตคอลทีซีพี/ไอพี",ซีเอ็ดยูเคชั่น,528 หน้า,2543.

[4] Deitel,"Java How to program",third edition,,Prentice Hall,1355 pages,2000.

[5] Jeff Doyle,"CCIE Professional Development:Routing TCP/IP",volume I,Macmillan Technical Publishing,1025pages,1998.

[6] Todd Lammle,Donald Porter with James Chellis,"CCNA Cisco Certified Network Associate Study Guide",729 pages,1999.

เว็บไซต์อ้างอิง

[1] http://www.cisco.com

[2] http://www.ku.ac.th/magazine_online

[3] http://www.ruencom.com/network.php

[4] http://www.kunkroo.com

[5] http://thaicert.nectec.or.th/paper/basic/vlan.php

[6] http://micro.se-ed.com

[7] http://pccare.cat.or.th

[8] http://student.cs.kku.ac.th

[9] http://learning.ricr.ac.th/datacomm/Subjectnew/Less5.html