บทที่ 2

ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

วิจัยเรื่องการจัดการเครือข่ายที่กำหนดโดยซอฟต์แวร์ โดยใช้เทคโนโลยีอินเทอร์เน็ตของสรรพสิ่ง ในการศึกษาครั้งนี้ผู้วิจัยได้ศึกษาค้นคว้า และงานวิจัยที่เกี่ยวข้องเพื่อนำมาใช้สำหรับการกำหนดกรอบความคิด หลักการ ทฤษฎี เครื่องมือ การรวบรวมข้อมูล การวิเคราะห์และการอภิปรายผลการศึกษา ซึ่งประกอบไปด้วยเนื้อหา ดังต่อไปนี้

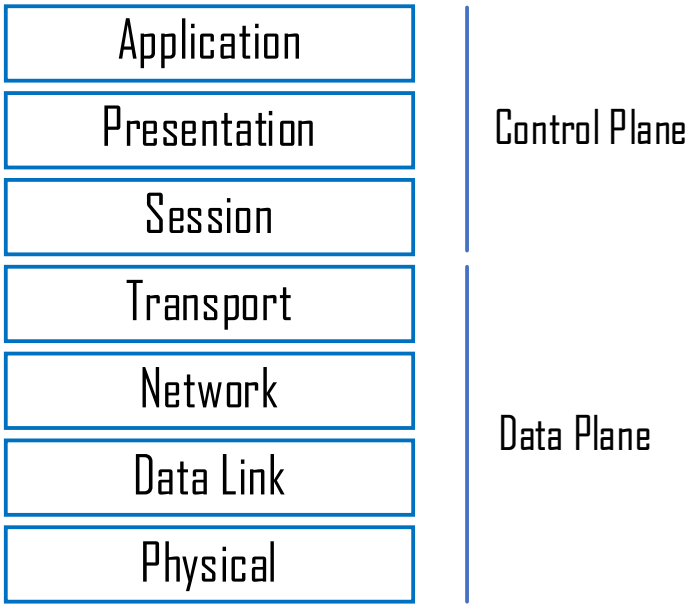
1. เครือข่ายที่กำหนดโดยซอฟต์แวร์ (software defined networking) SDN
2. รูปแบบการเข้าถึงอุปกรณ์เครือข่ายคอมพิวเตอร์ telnet / secure shell (SSH) / console
3. เอ็มคิวทีที (message queue telemetry transport) MQTT
4. อุปกรณ์เครือข่ายคอมพิวเตอร์
5. สแปนนิ่งทรีโปรโตคอล (spanning tree protocol) STP
6. เอ็นซิเบิล (ansible)
7. งานวิจัยที่เกี่ยวข้อง

2.1 เครือข่ายเครือข่ายที่กำหนดโดยซอฟต์แวร์ (software defined networking) SDN

ผู้ดูแลเครือข่าย มักมีปัญหากับการตั้งค่าอุปกรณ์ ที่จะมีความแตกต่างกันตามผู้ผลิต ทำให้ผู้ดูแลเครือข่ายไม่สามารถจัดการอุปกรณ์ได้อย่างมีประสิทธิภาพ ทำให้ภาระในการจัดการอุปกรณ์ที่แตกต่างกัน และจะต้องทำให้อุปกรณ์เครือข่ายสามารถทำงานร่วมกันได้ และส่วนใหญ่การจัดการอุปกรณ์เป็นไปในลักษณะ “ปิด” คือไม่มีวิธีการอื่นในการจัดการอุปกรณ์ให้สามารถทำงานร่วมกัน จึงจำเป็นต้องใช้วิธีการเข้าถึงอุปกรณ์เครือข่ายโดยtelnet / secure shell (SSH) และ console เพื่อที่เข้าไปจัดการแต่ละอุปกรณ์ได้ SDN เป็นวิธีการที่ถูกนำมาใช้เพื่อแก้ไขปัญหาการเข้าถึงอุปกรณ์เครือข่าย คือสถาปัตยกรรมของเครือข่าย ที่ดึงการจัดการเครือข่ายทั้งหมดมารวมอยู่ที่ซอฟต์แวร์ โดยไม่สนใจอุปกรณ์และความแตกต่างของอุปกรณ์เครือข่าย SDN สามารถแก้ไขปัญหาที่ผู้ดูแลเครือข่ายสามารถจัดการตั้งค่าอุปกรณ์ได้ในจุดเดียว แล้วอุปกรณ์เครือข่ายสามารถใช้งานได้ทันที เมื่อมีอุปกรณ์เครือข่ายใหม่ เพิ่มเข้ามาในระบบ ก็ไม่จำเป็นที่จะต้องแก้ไขที่ตัวอุปกรณ์เครือข่าย สามารถแก้ไขซอฟต์แวร์เพียงเล็กน้อย เพื่อกำหนดค่าอุปกรณ์เครือข่ายใหม่เท่านั้น

2.1.1 โครงสร้างของ SDN นั้นสร้างตัวเองอยู่บนฐานของแนวคิดที่ว่า ส่วนที่ควบคุม คอนโทรลเลอร์เพลนและส่วนของข้อมูล เดต้าเพลน แยกออกจากกันอย่างชัดเจน โดยมีส่วนของการจัดการ (management)

เป็นตัวเชื่อมโดยใช้ API ในการเชื่อมต่อทั้งสองส่วนนี้เข้าด้วยกัน คอนโทรลเลอร์เพลน เป็นส่วนที่จะทำหน้าที่ในการตัดสินใจว่า แพ็กเก็ตที่วิ่งอยู่ภายในระบบหรือเข้าถึงระบบแล้ว จะต้องจัดการส่งต่อหรือทำอย่างไรก็ต่อไป ส่วน เดต้าเพลน คือส่วนที่จะอนุญาต หรือทำหน้าที่ในการส่งข้อมูลไปตามการตัดสินใจของ คอนโทรลเลอร์เพลน เพื่อให้เห็นภาพชัดเจนขึ้นดู OSI Model เทียบกับแนวคิดดังกล่าว แสดงดังภาพที่ 2.1



ภาพที่ 2.1 OSI Model

แนวคิดดังกล่าวโดยในสภาพปัจจุบัน อุปกรณ์เครือข่ายแทบจะทุกอุปกรณ์จำเป็นที่จะต้องมี คอนโทรลเลอร์เพลน และ เดต้าเพลน อยู่ด้วยกันเสมอ ซึ่งแน่นอนว่าอาจจะต้องอยู่ในสภาพที่ปิดและต่างผู้ผลิต

ปัจจุบันอุปกรณ์ต่างๆ จะมีมาตรฐานบางอย่างกำหนดให้ทำงานร่วมกัน แต่สิ่งที่เกิดขึ้นก็คืออุปกรณ์แต่ละตัวมีการตัดสินใจจาก คอนโทรลเลอร์เพลน ที่แตกต่างกันโดยสิ้นเชิง และหลายครั้งจะพบว่า โปรโตคอลอย่าง TCP หรือ IP มักจะไม่ได้รับรองว่าข้อมูลที่ได้จะไปถึงปลายทางได้ (TCP ให้ความน่าเชื่อถือ ขณะที่ IP มีหน้าที่ควบคุม แต่กลับไม่มีความน่าเชื่อถือ) และหลายครั้งเมื่ออุปกรณ์หนึ่งส่งแพ็กเก็ตผิดพลาด ผลที่เกิดขึ้นคือแพ็กเก็ตเหล่านั้นต้องใช้ คอนโทรลเลอร์เพลน ของอุปกรณ์อื่นในการหาเส้นทางไปในแต่ละจุด (hop) ซึ่งสิ่งที่ตามมาคือการเสียเวลาโดยไม่จำเป็นที่การเดินทางของข้อมูลซึ่งสมควรจะเดินทางสั้นๆ แต่เมื่อเจอข้อผิดพลาดกลับต้องใช้ระยะทางในการเดินทำงานนานกว่าที่คิด ในกรณีแพ็กเก็ตที่ถูกส่งออกไป แม้ว่าจะทราบว่าปลายทางอยู่ที่ใด แต่การที่ต้องผ่านอุปกรณ์ต่างๆ ซึ่งทำหน้าที่อยู่ในเครือข่ายจำนวนมากในแต่ละจุด และหลายครั้งแต่ละอุปกรณ์มีระบบการควบคุมในการส่งต่อแพ็กเก็ตไปถึงปลายทางที่แตกต่างกัน ซึ่งทำให้การตัดสินใจในแต่ละจุดนั้นอาจจะไม่ได้หมายถึงว่า เป็นเส้นทางที่ดีที่สุดเสมอไป ถ้าเป็น SDN สิ่งที่เกิดขึ้นคือดึงเอา

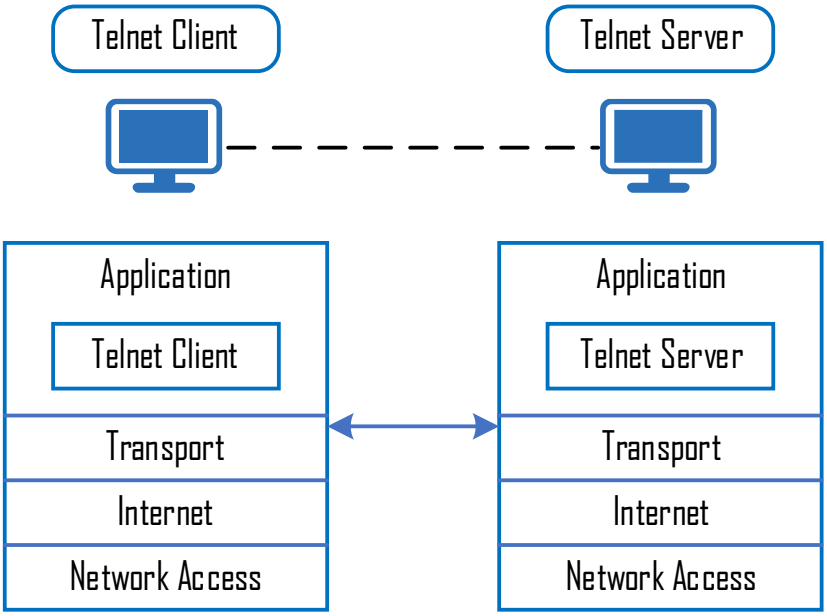
การทำงานในส่วนของ คอนโทรลเลอร์เพลนทั้งหมด ขึ้นมารวมไว้ที่จุดเดียว ผลที่ได้คือจะสามารถจัดการการเคลื่อนไหวของแพ็กเก็ตในเครือข่ายได้ และสามารถกำหนดหรือควบคุมโปรแกรม (program) เส้นทางของแต่ละแพ็กเก็ตโดยผ่านการกำหนดเงื่อนไขต่างๆ ผลที่เกิดขึ้นก็คือการจัดการเครือข่ายที่มีประสิทธิภาพได้ดีมากยิ่งขึ้น ลักษณะของ SDN ที่เน้นไปที่การควบคุมจากส่วนกลางและเส้นทางที่ดีที่สุดและสามารถเปลี่ยนแปลงเส้นทางให้สอดรับกับความหนาแน่นของเครือข่าย หมายความว่า ผู้ดูแลระบบสามารถเข้าจัดการเครือข่ายได้และกำหนดเส้นทางโดยใช้ซอฟต์แวร์ที่เรียกใช้งาน API ณ จุดเดียว ซึ่งทำให้สามารถใช้งานเครือข่ายได้อย่างเต็มที่และมีประสิทธิภาพ

2.2 รูปแบบการเข้าถึงอุปกรณ์เครือข่ายคอมพิวเตอร์

ในการจัดการอุปกรณ์เครือข่ายแต่ละชนิดเป็นเรื่องที่ยุ่งยากและซับซ้อน โดยอุปกรณ์เครือข่ายแต่ละผู้ผลิตจะมีวิธีการจัดการอุปกรณ์ของตัวเองที่แตกต่างกันออกไป แต่ในการจัดการอุปกรณ์เครือข่ายก็มีรูปแบบการเข้าถึงที่เป็นมาตรฐานอยู่ดังนี้

2.2.1 Console Port คือการกำหนดค่าอุปกรณ์เครือข่ายโดยใช้สาย Console Port ต่อระหว่างเครื่องคอมพิวเตอร์เข้ากับ Port Console ของอุปกรณ์เครือข่ายเพื่อที่จะสามารถเข้าไปจัดการกำหนดค่าอุปกรณ์เครือข่ายได้ มาตรฐานการเชื่อมต่อ Console Port คือ พอร์ตอนุกรม การเคลื่อนย้ายข้อมูลแบบอนุกรมนั้นเป็นการส่งข้อมูลครั้งละ 1 บิต ข้อดีของการสื่อสารข้อมูลแบบอนุกรมคือ สามารถส่งข้อมูลได้ในระยะทางที่ไกลและใช้สายสัญญาณที่น้อยกว่าการสื่อสารข้อมูลโดยปกติพอร์ตอนุกรม RS-232C จะสามารถต่อสายได้ยาว 50 ฟุตโดยประมาณ ขึ้นอยู่กับ ชนิดของ สายสัญญาณ, ระยะทาง, และ ปริมาณ สัญญาณ รบกวน เครื่องคอมพิวเตอร์ในปัจจุบันจะไม่มีการติดตั้งพอร์ตอนุกรมมาให้กับตัวเครื่องจึงต้องอาศัยอุปกรณ์ต่อพ่วงทำหน้าที่เป็นพอร์ตอนุกรมโดยต่อผ่านช่องทางพอร์ตชนิด USB แทน อุปกรณ์ชนิดนี้เรียกว่า USB To RS232

2.2.2 Telnet เป็นวิธีการติดต่อของผู้ใช้กับคอมพิวเตอร์เครื่องอื่น ที่ผู้ติดต่อได้รับอนุญาตในทางเทคนิค telnet เป็นคำสั่งของผู้ใช้ภายใต้โปรโตคอลTCP/IP เพื่อเข้าถึงคอมพิวเตอร์ในระยะไกล โปรโตคอลของเว็บ (Hypertext Transfer Protocol) และ File Transfer Protocol ยินยอมให้ผู้ใช้เรียกไฟล์ที่เจาะจงจากคอมพิวเตอร์ระยะไกล แต่ไม่ได้ logon ในฐานะผู้ใช้ของคอมพิวเตอร์เครื่องนั้น โดยการใช้ telnet ผู้ใช้สามารถ logon ในฐานะผู้ใช้ธรรมดาที่มีสิทธิในการเจาะจงโปรแกรมประยุกต์ และข้อมูลบนเครื่องคอมพิวเตอร์ ดังภาพ 2.2



ภาพที่ 2.2 Telnet server and client

โปรโตคอล Telnet ถูกกำหนดไว้ใน RFC 854 เป็นโปรโตคอลที่สามารถดำเนินการได้ทั้งฮาร์ดแวร์และซอฟต์แวร์ โดยวัตถุประสงค์พื้นฐานคือ ส่งค่าสั่งจากแป้นพิมพ์โดยผู้ใช้ไปยังปลายทาง การเชื่อมต่อไปยังเครื่องปลายทางด้วย Telnet แบ่งออกเป็น 2 ประเภท ดังนี้

1. Local Login เป็นการเชื่อมต่อกับ Server เพื่อเข้าใช้งานในระยะใกล้หรือเครือข่ายเดียวกัน โดยไม่ต้องอาศัยเครือข่ายอินเทอร์เน็ต

2. Remote Login เป็นการเชื่อมต่อเครื่องผู้ใช้กับแอปพลิเคชันภายในเครื่องคอมพิวเตอร์เครื่องอื่นที่อยู่ในระยะไกล โดยจะอาศัยเครือข่ายอินเทอร์เน็ตเป็นตัวกลางในการติดต่อกับเครื่องปลายทาง ผู้ใช้จำเป็นต้องทราบโดเมนเนม หรือไอพีเอสเดรสของเครื่องปลายทางด้วย

นอกจาก TELNET จะทำให้เราสามารถ login เข้าไปใช้งานเครื่องคอมพิวเตอร์เครื่องใดเครื่องหนึ่งแล้ว ยังสามารถ log in เข้าไปในเครื่องอื่นๆ ต่อได้อีกและสามารถเรียกใช้บริการต่างๆ บนเครื่องเหล่านั้นได้ด้วย โดยไม่จำเป็นต้องยกเลิกการติดต่อเครื่องคอมพิวเตอร์เครื่องแรก

2.2.3 Secure Shell ตัวย่อ SSH คือโปรแกรมสำหรับการเข้าถึงเครื่องคอมพิวเตอร์จากระยะไกล (remote login) แบบที่มีการคำนึงถึงความปลอดภัยผ่านโปรโตคอลเอสเอสเอช โดยมีการเข้ารหัสลับชื่อบัญชีผู้ใช้ (username) และ รหัสผ่าน (password) ที่จะส่งออกไป รวมทั้งสามารถเข้ารหัสลับในรูปแบบที่เป็นการถ่ายโอนไฟล์ (file transfer) ได้อีกด้วย พอร์ตมาตรฐานของเอสเอสเอช คือพอร์ต 22 ซึ่งจะต้องมีทฤษฎีและพื้นฐานเกี่ยวกับการเข้ารหัสลับมาเกี่ยวข้องด้วยเช่นกัน

- ในปี ค.ศ. 1995 โปรโตคอล SSH1 ได้ถูกพัฒนาขึ้นโดย Tatu Ylönen นักวิจัยแห่งมหาวิทยาลัยเทคโนโลยีเฮลซิงกิ ประเทศฟินแลนด์ และได้มีการก่อตั้งบริษัท SSH Communications Security, Ltd. ขึ้นในเดือนธันวาคม
- ในปี ค.ศ. 1996 บริษัท SSH Communications Security, Ltd. (SCS) ได้แนะนำโปรโตคอลรุ่นใหม่คือโปรโตคอล SSH2
- ในปี ค.ศ. 1998 บริษัท SSH Communications Security, Ltd. ได้เปิดตัวผลิตภัณฑ์ซอฟต์แวร์ “SSH Secure Shell”
- ในปี ค.ศ. 2000 บริษัท SSH Communications Security, Ltd. ได้เผยแพร่ใบอนุญาต SSH2 ให้ใช้ได้ฟรีสำหรับหน่วยงานที่ไม่ใช่เชิงพาณิชย์

SSH นั้นเป็นโปรแกรมชุดหนึ่งที่ใช้เชื่อมต่อกับเครื่องคอมพิวเตอร์ที่อยู่ไกลออกไปโดยผ่านทางเครือข่ายอินเทอร์เน็ต เพื่อที่จะดำเนินการคำสั่งบนเครื่องนั้น หรือเพื่อที่จะย้าย file จากเครื่องหนึ่งไปยังอีกเครื่องหนึ่ง SSH ทำให้เกิดมีการพิสูจน์ทราบถึงตัวจริงและช่วยให่การสื่อสารที่ปลอดภัยโดยใช้ระบบการเข้ารหัสลับที่แข็งแกร่งบนช่องการสื่อสารที่ไม่ปลอดภัย SSH ถูกเขียนขึ้นมาเพื่อใช้ทดแทนโปรแกรม telnet, rlogin, rsh, และ rcp ใน version ล่าสุดของ SSH คือ SSH2 นั้นมีโปรแกรมที่ใช้แทนโปรแกรม ftp เพิ่มขึ้นมาอีก นอกจากนี้ SSH ยังสามารถให้ความปลอดภัยแก่การเชื่อมต่อในรูปของ TCP

โปรโตคอล SSH จะทำงานอยู่บนโปรโตคอล TCP โดยจะถูกแบ่งออกเป็น 3 โปรโตคอลย่อยได้แก่

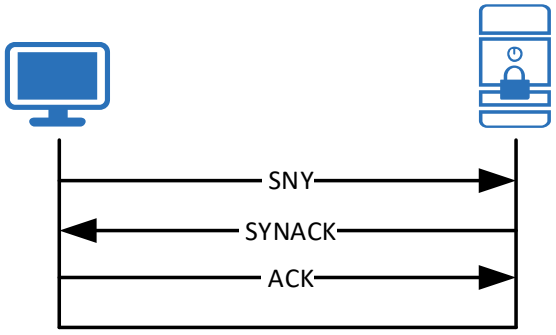
1. ชั้นทรานสปอร์ตโปรโตคอล: เตรียมไว้สำหรับ การตรวจสอบเซิร์ฟเวอร์, ปกป้องความลับของข้อมูล และ การคงสภาพของข้อมูล อาจจะเตรียมไว้สำหรับ การบีบอัดข้อมูล ซึ่งเป็นการลดขนาดของข้อมูลเพื่อประหยัดพื้นที่ หรือเวลาการในส่งข้อมูล ด้วย
2. โปรโตคอลการพิสูจน์ตัวจริงของผู้ใช้งาน : เป็นการพิสูจน์ตัวจริง สำหรับฝั่งผู้ใช้งานกับเซิร์ฟเวอร์
3. โปรโตคอลการเชื่อมต่อ : มีไว้สำหรับร่วมเส้นทางการสื่อสารแบบลอจิคอลหลายเส้นทางไปในเส้นทางพื้นฐานเส้นทางเดียว

SSH User **Authentication Protocol** Authentication the client-side User to the server	SSH **Connection Protocol** Multiplexes the encrypted tunnel into server logical channels
SSH Transport Layer Protocol Transmission control protocol provides reliable, connection – oriented end-to-end delivery.	
TCP Transmission control protocol provides reliable, connection – oriented end-to-end delivery.	
IP Internet protocol provides datagram delivery across multiple networks.	

ภาพที่ 2.3 SSH Protocol Stack

การแลกเปลี่ยนข้อมูล บน SSH ชั้นทรานสปอร์ตโปรโตคอลมีขั้นตอนการทำงานดังนี้

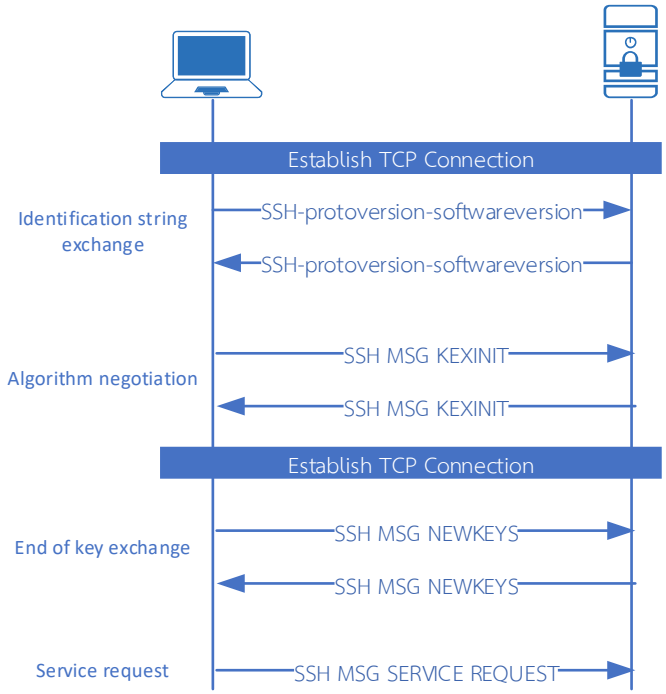
- Establish TCP Connection สร้างการเชื่อมต่อผ่านทางโปรโตคอลทีซีพี เครื่องไคลเอนต์ ทำการสร้างการเชื่อมต่อผ่านทางโปรโตคอลทีซีพี ไปยัง เครื่องเซิร์ฟเวอร์ หลังจากทำการเชื่อมต่อกันแล้ว เครื่องไคลเอนต์ ก็จะทำการแลกเปลี่ยนข้อมูลกับเครื่องเซิร์ฟเวอร์ โดยการใช้ Three-Way Handshake



ภาพที่ 2.4 การเชื่อมต่อแบบ Client Server

1. เครื่องไคลเอนต์ ส่ง Synchronization (SYN) ขอสร้างการเชื่อมต่อไปยังเครื่องเซิร์ฟเวอร์
2. เครื่องเซิร์ฟเวอร์ ตอบกลับโดยส่ง Synchronization and Acknowledgement (SYN, ACK) กลับไปยังเครื่อง client
3. เครื่องไคลเอนต์ ตอบกลับว่าได้รับแล้ว Acknowledgement (ACK) ไปทางเครื่องเซิร์ฟเวอร์ อีกครั้งหนึ่ง

- Identification String Exchange เป็นการแลกเปลี่ยนชื่อและรุ่นของซอฟต์แวร์ที่ให้บริการโปรโตคอล SSH
- Algorithm Negotiation คือการเจรจาเรื่องอัลกอริทึมที่จะใช้ โดยทั้งสองฝั่งจะทำการส่ง อัลกอริทึมระหว่างกัน หลังจากนั้นจะทำการเลือกอัลกอริทึมที่รองรับการใช้งานบนเครื่องเซิร์ฟเวอร์และ เครื่องไคลเอนต์ ซึ่งอัลกอริทึมหลักแบ่งออกเป็น 3 ส่วน
    1. Encryption การเข้ารหัสลับ เช่น AES128 CBC, 3DES-CBC, Blowfish-CBC, Cast128 CBC, Twofish-CBC, Arcfour
    2. Authentication การพิสูจน์ตัวจริง เช่น HMAC-MD5, HMAC-SHA1
    3. Compression การบีบอัดข้อมูล เช่น none (ไม่มีการใช้งาน), zlib
- Key Exchange การแลกเปลี่ยนคีย์ มีการใช้งานการแลกเปลี่ยนคีย์ชนิด Diffie-Hellman (DH) เป็นเทคนิคการกระจายและแลกเปลี่ยนเซสชั่นคีย์โดยใช้พารามิเตอร์สาธารณะ (Public parameter) และพารามิเตอร์ส่วนบุคคล (Private parameter) ซึ่งจะต้องเก็บพารามิเตอร์ส่วนบุคคลไว้เป็นความลับในขณะที่สามารถส่งพารามิเตอร์สาธารณะให้แก่ผู้อื่นได้ การกระจายคีย์ทำได้โดยการแลกเปลี่ยนพารามิเตอร์สาธารณะระหว่างกัน จากนั้นทั้งคู่จะนำพารามิเตอร์สาธารณะที่ได้รับมาคำนวณร่วมกับพารามิเตอร์ส่วนบุคคลที่ตนมีอยู่เพื่อสร้างเซสชั่นคีย์ขึ้นที่แต่ละฝ่าย ความพิเศษของวิธีการนี้คือ ต่างฝ่ายต่างสร้างเซสชั่นคีย์ที่มีค่าเหมือนกันได้โดยไม่มีการส่งคีย์นั้นผ่านเครือข่าย
- End of Key Exchange สิ้นสุดการแลกเปลี่ยนกุญแจ โดยทั้งสองฝ่ายจะแลกเปลี่ยนพารามิเตอร์สาธารณะซึ่งกัน
- Service Request ทางฝั่ง เครื่องไคลเอนต์ จะส่งแพ็กเกจร้องขอใช้งานเซอร์วิส ของ โปรโตคอลการพิสูจน์ตัวตนของผู้ใช้งาน หรือ โปรโตคอลการเชื่อมต่อ



ภาพที่ 2.5 SSH Transport Layer Protocol Packet Exchanges

จากภาพที่ 2.5 เมื่อช่องทางการเชื่อมต่อถูกสร้างขึ้นจนเกิดการแลกเปลี่ยนข้อมูลระหว่างเครื่องไคลเอนต์และเครื่องเซิร์ฟเวอร์ เราจะเรียกข้อมูลนี้ว่าแพ็กเกจ โดยจะอยู่ในส่วนพื้นที่ข้อมูลของทีซีพีเซกเมนต์ (data field of a TCP segment) ซึ่งมีรูปแบบดังนี้

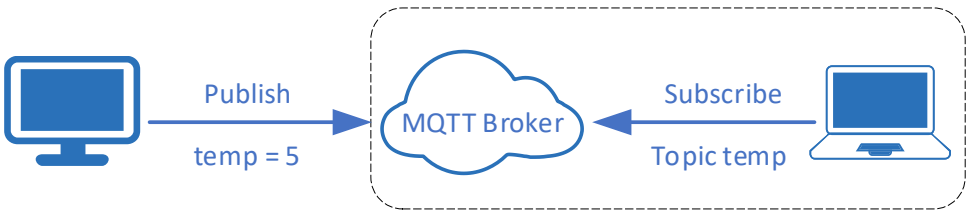
- Packet length: ความยาวของแพ็กเกจมีขนาดเป็นไบต์ ซึ่งไม่ได้รวมพื้นที่ความยาวของแพ็กเกจและพื้นที่รหัสพิสูจน์ตัวจริงข้อความ
- Padding length: ความยาวของแพดดิ้งในพื้นที่ของพื้นที่แพดดิ้งสุ่ม
- Payload: ข้อมูลที่ถือว่าเป็นเนื้อหาจริงของแพ็กเกจ
- Random padding: แพดดิ้งสุ่ม เป็นพื้นที่สุ่มเพื่อที่ขนาดของแพ็กเกจโดยร่วมจะได้ไม่แน่นอน
- Message Authentication Code (MAC): รหัสพิสูจน์ตัวจริงข้อความใช้ในการตรวจความคงสภาพของข้อมูล โดยจะมีการนำ sequence number ซึ่งเป็นพื้นที่ในโปรโตคอลทีซีพีมาร่วมใช้งานด้วย

2.3 เอ็มคิวทีที (Message Queue Telemetry Transport) MQTT

MQTT ย่อมาจาก Message Queuing Telemetry Transport เป็นโปรโตคอลสำหรับใช้ในสื่อสารข้อมูลระหว่าง Machine to Machine (M2M) ถูกคิดค้นขึ้นในปี ค.ศ. 1999 โดย Dr Andy Stanford-Clark จาก IBM และ Arlen Nipper จาก Arcom (now Eurotech) ออกแบบมาเพื่อใช้สื่อสารในระบบเครือข่ายที่มีทรัพยากรค่อนข้างจำกัด ใช้งานแบนด์วิดท์ต่ำ สามารถ publish-subscribe ข้อมูลระหว่าง Device เพื่อสื่อสารกันระหว่างอุปกรณ์ และถ้ามองในด้านที่เกี่ยวกับ Internet of Things จะสามารถประยุกต์ให้อุปกรณ์ต่างๆ เชื่อมต่อกันผ่านเครือข่ายของอินเทอร์เน็ตได้

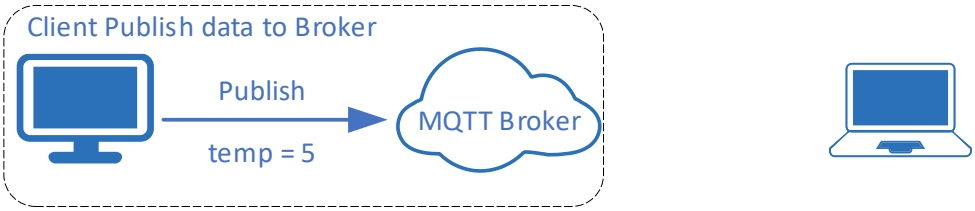
2.3.1 MQTT ประกอบด้วย ส่วนที่ 1 MQTT Client เป็นส่วน publish ข้อมูลต่างๆ ขึ้นไปยัง MQTT Broker และสามารถ Subscribe ข้อมูลต่างๆ จาก MQTT Broker ผ่านทาง TCP/IP Protocol ถ้ามองในมุมมองของ Internet of Things (IoT) อุปกรณ์จำพวกนี้จะเป็น Device ที่สามารถเชื่อมต่อกับระบบเครือข่ายได้ เช่น บอร์ด Arduino Uno Wifi 2, Arduino MKR Wifi 1010, บอร์ด ESP32, บอร์ด ESP8266, บอร์ด Raspberry Pi, เว็บไซต์, สมาร์ทโฟน ส่วนที่ 2 MQTT Broker หรือ MQTT Server เป็นซอฟต์แวร์สำหรับรับข้อมูลจาก MQTT Client ที่ได้ publish เข้ามาและสามารถ publish ข้อมูลจาก MQTT Broker ไปยัง MQTT Client ที่ได้ Subscribe ข้อมูลไว้ได้ หากมองในมุมมองของ Internet of Things อุปกรณ์นี้อาจจะเป็น Cloud Server ของค่ายต่างๆ เช่น CloudMQTT, NETPIE, Azure, AWS หรือใช้ Single Board Computer เช่น บอร์ด Raspberry Pi, LattePanda, Beagle Bone, nanoPi, อื่นๆ แล้วติดตั้งซอฟต์แวร์เพิ่มเติมก็สามารถใช้งานได้เช่นกัน

2.3.2 ตัวอย่างขั้นตอนการสื่อสารของ MQTT ขั้นตอนที่ 1 กำหนดให้ Client Device เป็นสมาร์ทโฟน ทำการ Subscribe MQTT Broker ตาม Topic ที่ต้องการ คือ temp ไว้ ดังภาพที่ 2.6



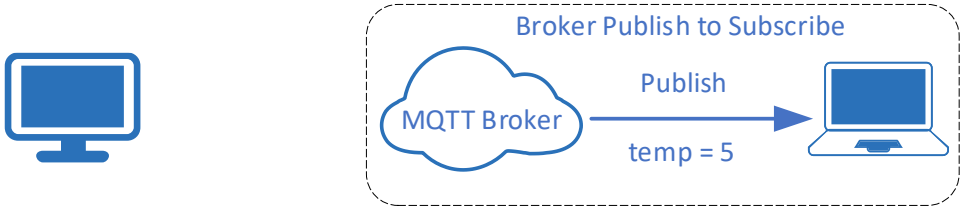
ภาพที่ 2.6 รูปแบบการทำงานของ MQTT

ขั้นตอนที่ 2 กำหนดให้ Client Device (Device ที่ต่อกับเซ็นเซอร์วัดอุณหภูมิ) แล้วทำการ Publish ค่าขึ้นไปยัง MQTT Broker ดังภาพที่ 2.7



ภาพที่ 2.7 การทำงานของ MQTT โดย Client Publish ไปยัง MQTT Broker

ขั้นตอนที่ 3 MQTT Broker Publish ไปยังอุปกรณ์ที่ Subscribe ไว้ ดังนั้นในตัวอย่างนี้ไคลเอนต์จะได้รับข้อมูลจาก Client ที่ต่ออยู่กับเซ็นเซอร์เรียบร้อย ดังภาพที่ 2.8



ภาพที่ 2.8 การทำงานของ MQTT โดย Broker Publish ไปยัง Client ปลายทาง

จากตัวอย่างการสื่อสารเบื้องต้นของ MQTT จะเห็นได้ว่าการสื่อสารระหว่างอุปกรณ์นั้นสามารถเชื่อมต่อกันได้อย่างกว้างขวาง ไม่จำกัดในเรื่องชนิดอุปกรณ์ว่าจำเป็นต้องเป็นอุปกรณ์ชนิดเดียวกัน สามารถสื่อสารกันได้หลากหลายแพลตฟอร์ม มีตัวอย่างการใช้งานสำหรับการพัฒนาโปรแกรมด้วยภาษาต่างๆ มากมายเช่น ภาษา C, Python

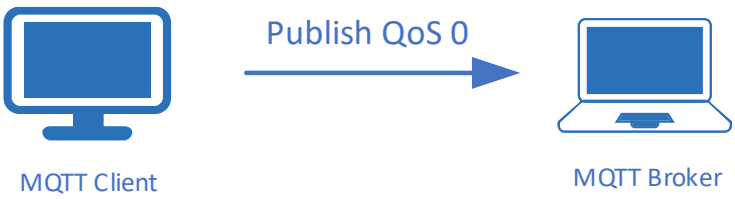
2.3.3 เอ็มคิวทีทีคิวโอเอส (MQTT QoS) ระดับคุณภาพของบริการ (QoS) เป็นข้อตกลงระหว่างผู้ส่งข้อความและผู้รับข้อความที่กำหนดการรับประกันการส่งมอบสำหรับข้อความเฉพาะ คิวโอเอา (QoS) ใน เอ็มคิวทีที (MQTT) มี 3 ระดับดังนี้

- คิวโอเอสศูนย์ (Qos 0)
- คิวโอเอสศูนย์ (Qos 1)
- คิวโอเอสศูนย์ (Qos 2)

คิวโอเอส (QoS) ใน เอ็มคิวทีที (MQTT) ต้องพิจารณาการส่งข้อความทั้งสองด้าน การส่งข้อความจากไคลเอนต์ (Client) ที่เผยแพร่ไปยังเซิร์ฟเวอร์ (Server) และการส่งข้อความจาก เซิร์ฟเวอร์ (Server) ไปยังไคลเอนต์ (Client) พิจารณาทั้งสองด้านของการส่งข้อความแยกกัน เนื่องจากมีความแตกต่างกันเล็กน้อย

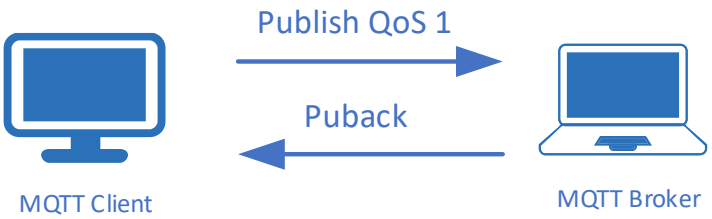
ระหว่างทั้งสอง ไคลเอนต์ (Client) ที่เผยแพร่ข้อความไปยัง เซิร์ฟเวอร์ (Server) กำหนดระดับ QoS ของข้อความเมื่อส่งข้อความไปยังเซิร์ฟเวอร์ (Server) โบรกเกอร์ส่งข้อความนี้ไปยัง ไคลเอนต์ (Client) ที่สมัครรับข้อมูลโดยใช้ระดับ QoS ที่ ไคลเอนต์ (Client) ที่สมัครรับข้อมูลแต่ละรายกำหนดในระหว่างกระบวนการสมัครสมาชิก หากไคลเอนต์ (Client) ที่สมัครรับข้อมูลกำหนด QoS ที่ต่ำกว่า ไคลเอนต์ (Client) ที่เผยแพร่เซิร์ฟเวอร์ (Server) จะส่งข้อความด้วยคุณภาพการบริการที่ต่ำกว่า QoS เป็นคุณสมบัติหลักของโปรโตคอลเอ็มคิวทีที (MQTT) คิวโอเอส (QoS) ช่วยให้ ไคลเอนต์ (Client) สามารถเลือกระดับของบริการที่ตรงกับความน่าเชื่อถือของเครือข่ายและตรรกะของแอปพลิเคชัน เนื่องจาก เอ็มคิวทีที (MQTT) จัดการการส่งข้อความซ้ำและรับประกันการส่ง (แม้ว่าการส่งข้อมูลพื้นฐานไม่น่าเชื่อถือ) คิวโอเอส (QoS) ทำให้การสื่อสารในเครือข่ายที่ไม่น่าเชื่อถือง่ายขึ้นมาก เอ็มคิวทีที คิวโอเอส (MQTT Qos) ทำงานอย่างไร การทำงานแต่ละระดับ คิวโอเอส (QoS) ถูกนำไปใช้ในโปรโตคอลเอ็มคิวทีที (MQTT) อย่างไรและทำงานอย่างไรดังตัวอย่างต่อไปนี้

1. คิวโอเอสศูนย์ (QoS 0) ระดับ QoS ขั้นต่ำคือศูนย์ ระดับการบริการนี้รับประกันการส่งมอบอย่างดีที่สุด ไม่มีการรับประกันการจัดส่ง ผู้รับไม่รับทราบการรับข้อความและข้อความจะไม่ถูกจัดเก็บและส่งต่อโดยผู้ส่ง QoS ระดับ 0 มักเรียกว่า (Fire And Forget) และให้การรับประกันเช่นเดียวกับโปรโตคอล TCP พื้นฐาน



ภาพที่ 2.9 คิวโอเอสศูนย์ (QoS 0)

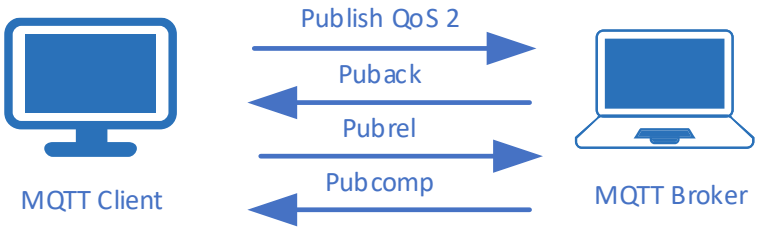
2. คิวโอเอสหนึ่ง (QoS 1) ระดับ 1 รับประกันว่าข้อความถูกส่งไปยังผู้รับอย่างน้อยหนึ่งครั้ง ผู้ส่งเก็บข้อความไว้จนกว่าจะได้รับแพ็กเก็ต PUBACK จากผู้รับที่รับทราบการรับข้อความ เป็นไปได้ที่จะส่งหรือส่งข้อความหลายครั้ง



ภาพที่ 2.10 คิวโอเอสศูนย์ (QoS 1)

ผู้ส่งใช้ตัวระบุแพ็กเก็ตในแต่ละแพ็กเก็ตเพื่อจับคู่แพ็กเก็ต PUBLISH กับแพ็กเก็ต PUBACK ที่เกี่ยวข้อง หากผู้ส่งไม่ได้รับแพ็กเก็ต PUBACK ในระยะเวลาที่เหมาะสม ผู้ส่งจะส่งแพ็กเก็ต PUBLISH อีกครั้ง เมื่อผู้รับได้รับข้อความที่มี QoS 1 ก็สามารถประมวลผลได้ทันที ตัวอย่างเช่น หากผู้รับเป็นนายหน้า นายหน้าจะส่งข้อความไปยังลูกค้าที่สมัครรับข้อมูลทั้งหมดแล้วตอบกลับด้วยแพ็กเก็ต PUBACK หากไคลเอนต์การเผยแพร่ส่งข้อความอีกครั้ง จะตั้งค่าสถานะซ้ำ (DUP) ใน QoS 1 การตั้งค่าสถานะ DUP นี้ใช้เพื่อวัตถุประสงค์ภายในเท่านั้น และไม่ได้รับการประมวลผลโดยนายหน้าหรือลูกค้า ผู้รับข้อความจะส่ง PUBACK โดยไม่คำนึงถึงการตั้งค่าสถานะ DUP

3. คิวโอเอสสอง (QoS 2) เป็นบริการระดับสูงสุดใน MQTT ระดับนี้รับประกันว่าแต่ละข้อความจะได้รับเพียงครั้งเดียวโดยผู้รับที่ต้องการ QoS 2 เป็นระดับบริการที่มีคุณภาพที่ปลอดภัยและช้าที่สุด การรับประกันจัดทำโดยขั้นตอนการร้องขอ/การตอบสนองอย่างน้อยสองครั้ง (การจับมือสี่ส่วน) ระหว่างผู้ส่งและผู้รับ ผู้ส่งและผู้รับใช้ตัวระบุแพ็กเก็ตของข้อความ PUBLISH ดั้งเดิมเพื่อประสานงานการส่งข้อความ



ภาพที่ 2.11 คิวโอเอสศูนย์ (QoS 2)

เมื่อผู้รับได้รับแพ็กเก็ต QoS 2 PUBLISH จากผู้ส่ง ระบบจะประมวลผลข้อความเผยแพร่ตามนั้นและตอบกลับผู้ส่งด้วยแพ็กเก็ต PUBREC ที่ยอมรับแพ็กเก็ต PUBLISH หากผู้ส่งไม่ได้รับแพ็กเก็ต PUBREC จากผู้รับ ระบบจะส่งแพ็กเก็ต PUBLISH อีกครั้งพร้อมแฟล็กซ้ำ (DUP) จนกว่าจะได้รับการตอบรับ เมื่อผู้ส่งได้รับแพ็กเก็ต PUBREC จากผู้รับ ผู้ส่งสามารถละทิ้งแพ็กเก็ต PUBLISH เริ่มต้นได้อย่างปลอดภัย ผู้ส่งเก็บแพ็กเก็ต PUBREC จากผู้รับและตอบกลับด้วยแพ็กเก็ต PUBREL หลังจากที่ผู้รับได้รับแพ็กเก็ต PUBREL ก็สามารถละทิ้งสถานะที่เก็บไว้ทั้งหมดและตอบด้วยแพ็กเก็ต PUBCOMP (เช่นเดียวกันเมื่อผู้ส่งได้รับ PUBCOMP) จนกว่าผู้รับจะประมวลผลเสร็จและส่งแพ็กเก็ต PUBCOMP กลับไปยังผู้ส่ง ผู้รับจะเก็บข้อมูลอ้างอิงไปยังตัวระบุแพ็กเก็ตของแพ็กเก็ต PUBLISH ดั้งเดิม ขั้นตอนนี้มีความสำคัญเพื่อหลีกเลี่ยงการประมวลผลข้อความเป็นครั้งที่สอง หลังจากที่ผู้ส่งได้รับแพ็กเก็ต PUBCOMP แล้ว ตัวระบุแพ็กเก็ตของข้อความที่เผยแพร่จะพร้อมใช้ซ้ำได้ เมื่อโฟลว์ QoS 2 เสร็จสมบูรณ์ ทั้งสองฝ่ายจะแน่ใจว่าข้อความถูกส่งไปแล้ว และผู้ส่งได้รับการยืนยันการจัดส่งแล้ว หากแพ็กเก็ตสูญหายระหว่างทาง ผู้ส่งจะต้องรับผิดชอบในการส่งข้อความซ้ำภายในระยะเวลาที่

เหมาะสม สิ่งนี้เป็นจริงเท่าเทียมกันหากผู้ส่งเป็นไคลเอนต์ MQTT หรือโบรกเกอร์ MQTT ผู้รับมีหน้าที่ตอบกลับข้อความคำสั่งแต่ละข้อความตามลำดับ

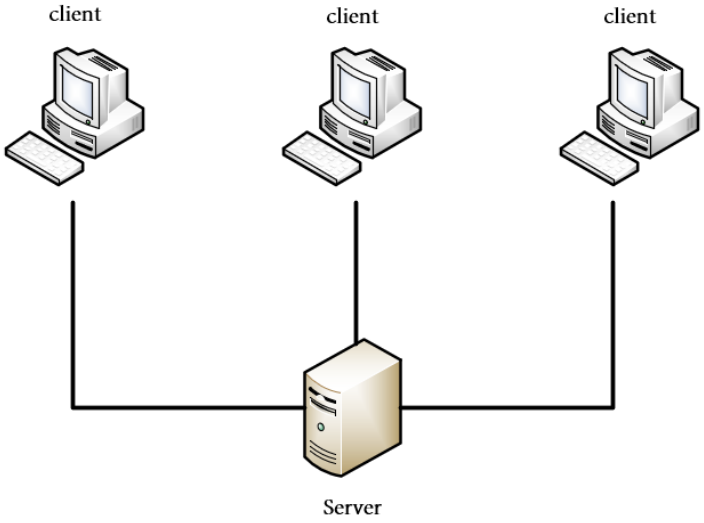
4. การใช้งาน (QoS 0) มีการเชื่อมต่อระหว่างผู้ส่งและผู้รับอย่างสมบูรณ์หรือเป็นส่วนใหญ่ กรณีการใช้งานแบบคลาสสิกสำหรับ QoS 0 กำลังเชื่อมต่อไคลเอนต์ทดสอบหรือแอปพลิเคชันส่วนหน้ากับโบรกเกอร์ MQTT ผ่านการเชื่อมต่อแบบมีสาย ไม่สนใจหากข้อความบางข้อความหายไปในบางครั้ง การสูญเสียข้อความบางข้อความสามารถยอมรับได้หากข้อมูลไม่สำคัญหรือเมื่อข้อมูลถูกส่งในช่วงเวลาสั้นๆ ไม่จำเป็นต้องเข้าคิวข้อความ ข้อความจะถูกจัดคิวสำหรับไคลเอนต์ที่ตัดการเชื่อมต่อหากมี QoS 1 หรือ 2 และเซสชันถาวร

5. การใช้งาน (QoS 1) ต้องได้รับทุกข้อความและกรณีการใช้งานของคุณสามารถจัดการกับรายการซ้ำได้ QoS ระดับ 1 เป็นระดับบริการที่ใช้บ่อยที่สุดเพราะรับประกันว่าข้อความจะมาถึงอย่างน้อยหนึ่งครั้ง แต่อนุญาตให้มีการจัดส่งหลายครั้ง แน่นอน ใบสมัครต้องทนต่อการทำซ้ำและสามารถดำเนินการตามนั้นได้ ไม่สามารถแบกรับค่าใช้จ่ายของ QoS 2 ได้ QoS 1 ส่งข้อความได้เร็วกว่า QoS 2 มาก

6. การใช้งาน (QoS 2) การสมัครต้องได้รับข้อความทั้งหมดเพียงครั้งเดียว กรณีนี้มักเกิดขึ้นหากการส่งซ้ำอาจเป็นอันตรายต่อผู้ใช้แอปพลิเคชันหรือไคลเอนต์ที่สมัครรับข้อมูล ระวังค่าใช้จ่ายและการโต้ตอบ QoS 2 จะใช้เวลามากขึ้นในการดำเนินการให้เสร็จสิ้น

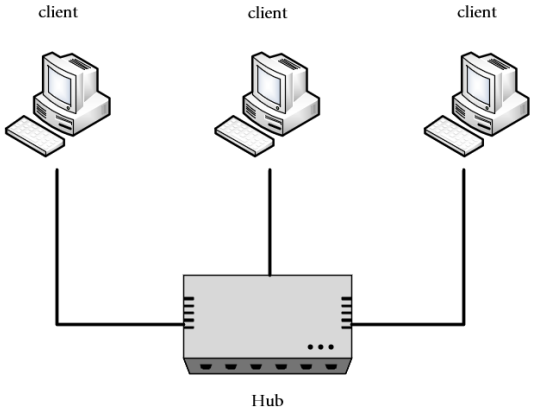
2.4 อุปกรณ์เครือข่ายคอมพิวเตอร์

เซอร์เวอร์ (Server) หรือเรียกอีกอย่างหนึ่งว่า เครื่องแม่ข่าย เป็นเครื่องคอมพิวเตอร์หลักในเครือข่าย ที่ทำหน้าที่จัดเก็บและให้บริการไฟล์ข้อมูลและทรัพยากรอื่นๆ กับคอมพิวเตอร์เครื่องอื่นๆ ใน เครือข่าย โดยปกติคอมพิวเตอร์ที่นำมาใช้เป็นเซิร์ฟเวอร์มักจะเป็นเครื่องที่มีสมรรถนะสูง และมีฮาร์ดดิสก์ความจำสูงกว่าคอมพิวเตอร์เครื่องอื่นๆ ในเครือข่าย ดังรูปภาพที่ 2.12



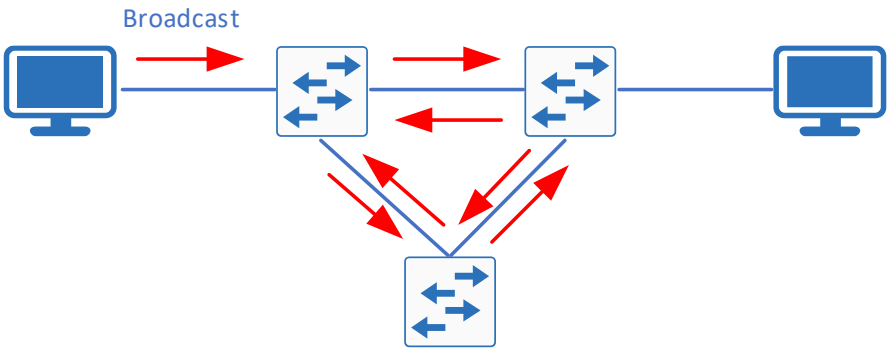
ภาพที่ 2.12 การเชื่อมต่อระหว่างไคลเอนต์และเซิร์ฟเวอร์

ฮับ (HUB) หรือ เรียก รีพีตเตอร์ (Repeater) คืออุปกรณ์ที่ใช้เชื่อมต่อกลุ่มคอมพิวเตอร์ ฮับ มีหน้าที่รับส่งเฟรมข้อมูลทุกเฟรมที่ได้รับจากพอร์ตใดพอร์ตหนึ่ง ไปยังพอร์ตที่เหลือ คอมพิวเตอร์ที่เชื่อมต่อเข้ากับฮับจะแชร์แบนด์วิดท์หรืออัตราข้อมูลของเครือข่าย เพราะฉะนั้นถ้ามีคอมพิวเตอร์เชื่อมต่อมากจะทำให้อัตราการส่งข้อมูลลดลงฮับเปรียบเสมือนเป็นบัสที่รวมอยู่ที่จุดเดียวกัน ฮับที่ใช้งานอยู่ภายใต้มาตรฐานการรับ-ส่งแบบอีเทอร์เน็ต หรือ IEEE802.3 ข้อมูลที่รับ-ส่งผ่านฮับจากเครื่องหนึ่งกระจายไปยังทุกสถานีที่ติดต่ออยู่บนฮับนั้น ดังนั้นทุกสถานีจะรับสัญญาณข้อมูลที่กระจายมาได้ทั้งหมดแต่จะเลือกคัดลอกเฉพาะข้อมูลที่ส่งมาถึงตนเท่านั้น การตรวจสอบข้อมูลจึงต้องดูที่แอดเดรส (Address) ที่กำกับมาในกลุ่มของข้อมูลหรือแพ็กเกจ



ภาพที่ 2.13 การเชื่อมต่อระหว่างไคลเอนต์กับฮับ

สวิตช์ (Switch) คืออุปกรณ์เครือข่ายที่ทำหน้าที่ในเลเยอร์ที่ 2 และในปัจจุบันมีถึงเลเยอร์ที่ 3 ทำหน้าที่ส่งข้อมูลที่ได้รับมาจากพอร์ตหนึ่งไปยังพอร์ตเฉพาะที่เป็นปลายทางเท่านั้น และทำให้คอมพิวเตอร์ที่เชื่อมต่อกับพอร์ตที่เหลือส่งข้อมูลถึงกันในเวลาเดียวกัน ดังนั้น อัตราการรับส่งข้อมูลหรือแบนด์วิดท์จึงไม่ขึ้นอยู่กับคอมพิวเตอร์ ปัจจุบันนิยมเชื่อมต่อแบบนี้มากกว่าฮับเพราะลดปัญหาการชนกันของข้อมูล ข้อแตกต่างจากฮับ คือ การรับ-ส่งข้อมูลจากสถานีหรืออุปกรณ์ตัวหนึ่งจะไม่กระจายไปยังทุกสถานีเหมือนฮับ ทั้งนี้เพราะสวิตช์จะรับกลุ่มข้อมูลหรือแพ็กเกจมาตรวจสอบก่อน แล้วดูว่าแอดเดรสของสถานีปลายทางไปที่ใด สวิตช์จะลดปัญหาการชนกันของข้อมูลเพราะ ไม่ต้องกระจายข้อมูลไปทุกสถานี และยังมีข้อดีในเรื่องการป้องกันการดักจับข้อมูลที่กระจายไปในเครือข่าย ดังภาพที่ 2.14



ภาพที่ 2.14 การทำงานของสวิตช์ (Switch)

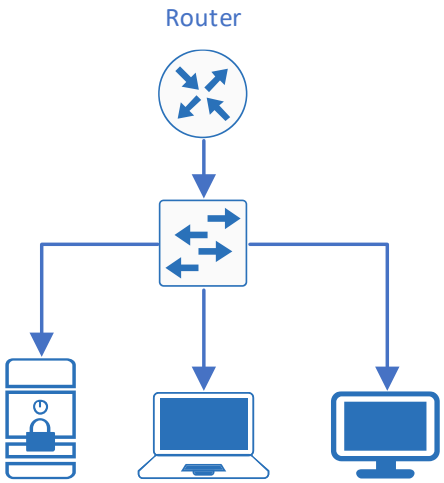
ความแตกต่างระหว่างสวิตช์เลเยอร์ 2 และ สวิตช์เลเยอร์ 3 ในการส่งข้อมูลแบบ Switch layer 2 นั้นจะพิจารณาจาก MAC address เป็นหลักและมีการทำตารางเพื่อเก็บข้อมูล MAC address ของอุปกรณ์ที่เชื่อมต่อกันอยู่ และเมื่อมีการส่งข้อมูลหากัน Switch จะนำ MAC address มาเทียบกับตารางเพื่อดูว่าเป็นของ port ไหนและทำการส่งข้อมูลไปยัง port นั้นๆ ต่างจาก Hub ที่ทำการกระจายข้อมูลออกไปทุกพอร์ต

Switch Layer 3 สามารถทำงานของ Layer 2 ได้เมื่อเป็นการส่งข้อมูลในระดับ Layer 2 และมีคุณสมบัติในการทำ static routing และ dynamic routing

- Static routing คือการกำหนดเส้นทางแบบตายตัวทำให้ router ไม่ต้องหาเส้นทางอื่นทำให้ข้อมูลมีความปลอดภัยสูง
- Dynamic routing คือการคำนวณหาเส้นทางที่สั้นที่สุดด้วยตัวมันเอง และถ้ามีเส้นทางใดถูกตัดขาด Router จะค้นหาเส้นทางอื่นมาทดแทนให้

หมายความว่า Layer3 จะทำได้ทั้งตาราง MAC address และตาราง IP routing และสามารถสื่อสารระหว่าง VLAN ได้ ส่วน Layer2+ หรือ Layer 3 lite จะสามารถทำได้แค่ static routing และถึงแม้ว่า layer 3 จะมีความสามารถในการ Routing ได้เหมือนกับ Router แต่จะไม่สามารถ broad cast ข้ามเครือข่ายและไม่สามารถทำ NAT (Network Address Translation) ได้

เราต์เตอร์ (Router) เป็นอุปกรณ์ที่ทำหน้าที่ในเลเยอร์ที่ 3 เราท์เตอร์จะอ่านที่อยู่ (Address) ของสถานีปลายทางที่ส่วนหัว (Header) ข้อแพ็กเก็ตข้อมูล เพื่อที่จะกำหนดและส่งแพ็กเก็ตต่อไป เราท์เตอร์จะมีตัวจัดเส้นทางในแพ็กเก็ต เรียกว่า เร้าติ้งเทเบิล (Routing Table) หรือตารางจัดเส้นทางนอกจากนี้ยังส่งข้อมูลไปยังเครือข่ายที่ให้โพรโทคอลต่างกันได้ เช่น IP (Internet Protocol) , IPX (Internet Package Exchange) และ AppleTalk นอกจากนี้ยังเชื่อมต่อกับเครือข่ายอื่นได้ เช่น เครือข่ายอินเทอร์เน็ต ดังรูปภาพที่ 2.15

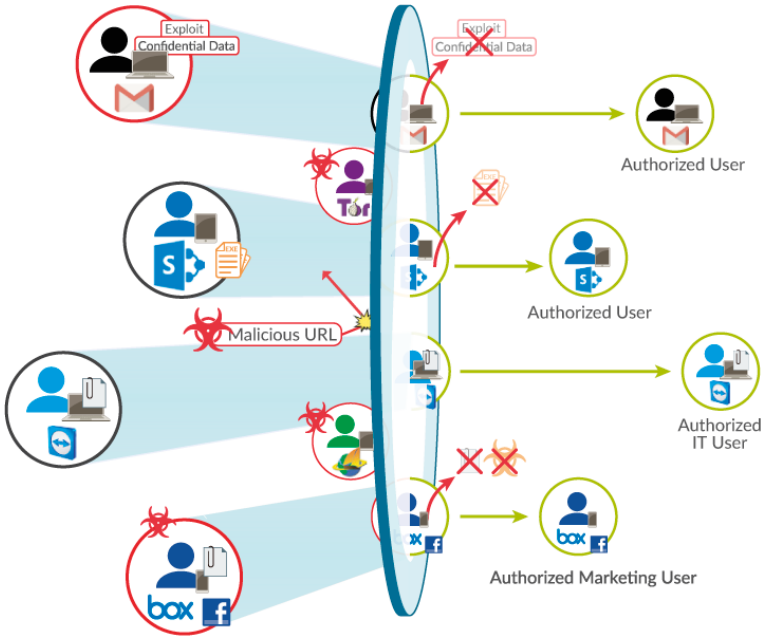


ภาพที่ 2.15 ลักษณะการทำงานของ Router

ไฟร์วอลล์ (Firewall) คืออุปกรณ์รักษาความปลอดภัยเครือข่ายที่ตรวจสอบการรับส่งข้อมูลเครือข่ายขาเข้าและขาออกและตัดสินใจว่าจะอนุญาตหรือบล็อกการรับส่งข้อมูลที่ระบุตามกฎความปลอดภัยที่กำหนดไว้หรือไม่ ชนิดของไฟร์วอลล์

- ไฟร์วอลล์พร็อกซี ชนิดเริ่มต้นของอุปกรณ์ไฟร์วอลล์พร็อกซีไฟร์วอลล์ทำหน้าที่เป็นเกตเวย์จากเครือข่ายหนึ่งไปยังอีกสำหรับโปรแกรมประยุกต์ที่ระบุ พร็อกซีเซิร์ฟเวอร์สามารถให้การทำงานเพิ่มเติมเช่นเนื้อหาแคชและการรักษาความปลอดภัยโดยการป้องกันการเชื่อมต่อโดยตรงจากภายนอกเครือข่าย อย่างไรก็ตามนี้ยังอาจส่งผลกระทบต่อความสามารถในการผลิตและโปรแกรมที่พวกเขาสามารถสนับสนุนได้

- ไฟร์วอลล์การตรวจสอบ Stateful ตอนนี้คิดว่า "แบบดั้งเดิม" ไฟร์วอลล์การตรวจสอบ stateful ช่วยให้หรือบล็อกการจราจรขึ้นอยู่กับรัฐพอร์ตและโพรโทคอล มันตรวจสอบกิจกรรมทั้งหมดจากการเปิดการเชื่อมต่อจนกว่าจะถูกปิด การตัดสินใจในการกรองจะขึ้นอยู่กับทั้งกฎที่กำหนดโดยผู้ดูแลระบบเช่นเดียวกับบริบทซึ่งอ้างอิงถึงการใช้ข้อมูลจากการเชื่อมต่อก่อนหน้าและแพคเก็ตที่อยู่ในการเชื่อมต่อเดียวกัน
- ไฟร์วอลล์แบบรวมการจัดการ (UTM) อุปกรณ์ UTM โดยทั่วไปจะรวมกันในวิธีการที่มีคู่หลวมฟังก์ชันของไฟร์วอลล์การตรวจสอบ stateful ด้วยการป้องกันการบุกรุกและการป้องกันไวรัส นอกจากนี้ยังอาจมีบริการเพิ่มเติมและมักจะจัดการคลาวด์ UTMs เน้นความเรียบง่ายและใช้งานง่ายการรักษาความปลอดภัยและการควบคุมเครือข่ายภายในที่สามารถเชื่อถือได้และมีการป้องกันไว้นอกเครือข่ายเช่นอินเทอร์เน็ตไฟร์วอลล์อาจเป็นฮาร์ดแวร์ซอฟต์แวร์หรือทั้งสองอย่าง ดังภาพที่ 2.16



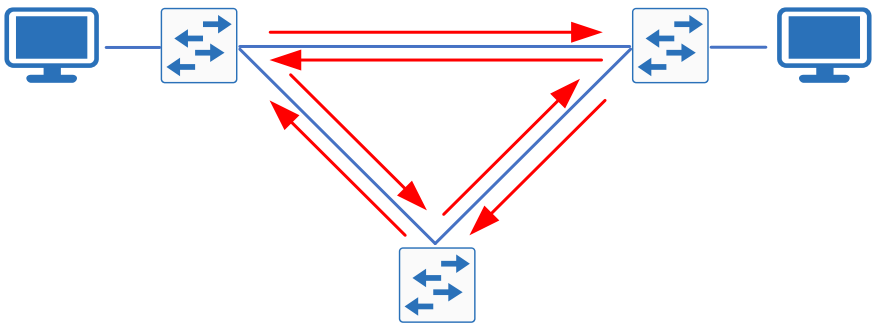
ภาพที่ 2.16 หลักการทำงานของ Firewall

(ที่มา: http://blog.onestopware.com/wp-content/uploads/2017/02/NGFW.png)

2.5 สแปนนิ่งทรีโปรโตคอล (STP - Spanning Tree Protocol)

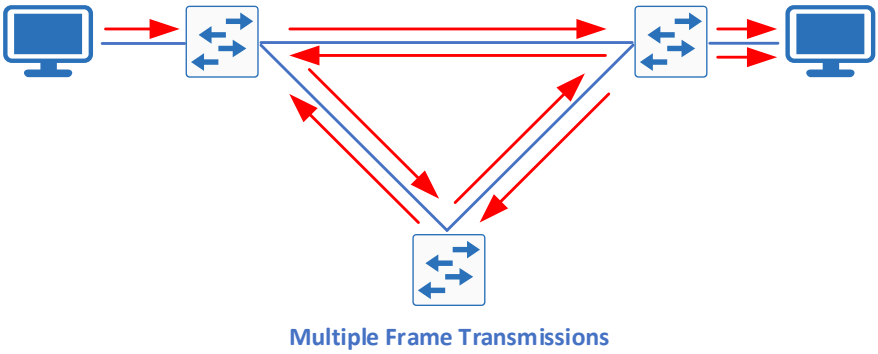
การเกิด Loop ในระบบเครือข่ายนั้นเป็นปัญหาร้ายแรงอย่างมาก เพราะจะทำให้การรับส่งข้อมูลนั้นมีปัญหาและทำให้ระบบไม่สามารถสื่อสารกันได้ดังนั้นในการติดตั้งระบบเครือข่าย ผู้ติดตั้งระบบจำเป็นต้องมีทักษะและวิธีการรับแก้ไขปัญหาการเกิด Loop ในระบบเครือข่ายนี้ และวิธีที่นิยมใช้ในการแก้ไขปัญหานี้ก็คือการทำ Spanning Tree ในระบบเครือข่าย

2.5.1. ลูป Loop คือ สถานะที่อุปกรณ์ส่งข้อมูลออกไปบนระบบ แล้วเกิดการวนแบบไม่มีที่สิ้นสุด เมื่อมีการต่อใช้งานระบบเครือข่ายผ่าน Switch โดยมีเส้นทางเดียวจากต้นทางไปยังปลายทาง ถ้า Link ระหว่าง Switch ทั้งสองตัวเกิดมีปัญหาขึ้นมา ก็ทำให้ระบบไม่สามารถใช้งานได้ เราจึงต้องมีการเพิ่ม Switch ขึ้นมาอีกหนึ่งตัว เพื่อให้มีเส้นทางสำรอง เมื่อ Link ระหว่าง Switch เดิมมีปัญหา ระบบจะยังสามารถใช้งานต่อได้ โดยผ่านไปทาง Switch อีกตัว แต่ว่าเนื่องจาก Switch ทั้ง 3 ตัว ต่อกันแบบ Loop ทำให้ข้อมูลที่ส่งไป เกิดการส่งวนไปวนมาได้ แบบไม่มีที่สิ้นสุด โดยทางเทคนิคเราเรียกปัญหานี้ว่าการเกิด Bridge Loop ซึ่งหลังจากการเกิด Bridege Loop มาแล้ว จะมีปัญหาหลักๆ อยู่ 3 อย่างตามมาในระบบเครือข่าย ดังต่อไปนี้ Broadcast Storm คือเมื่อเกิด Loop ขึ้น Broadcast Traffic จะถูกส่งต่อไปเรื่อยๆ แบบไม่สิ้นสุด เกิดเป็นพายุBroadcast (Broadcast Storm) ทำให้ CPU ของ Switch นั้นสูงขึ้นและทำงานไม่ได้ในที่สุด ดังภาพที่ 2.17



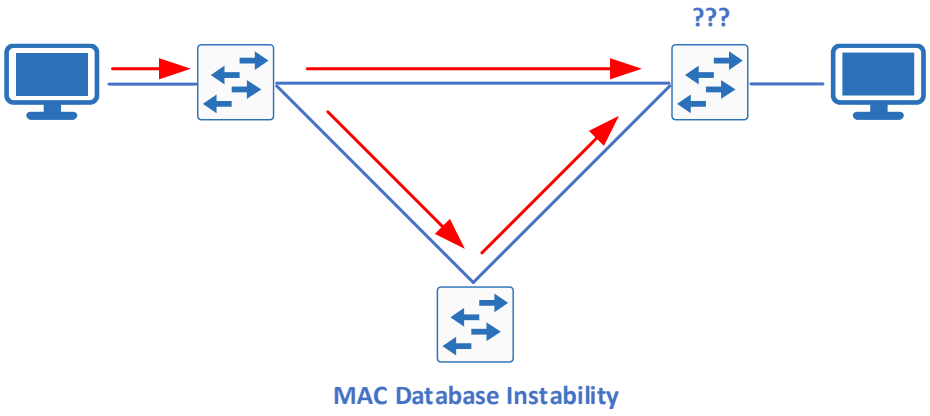
ภาพที่ 2.17 บรอดแคสสตรอม

Multiple Frame Transmissions คือเครื่องปลายทางได้รับข้อมูล (Frame) เข้ามาซ้ำ ทำให้เสียเวลาในการประมวลผล ดังภาพที่ 2.18



ภาพที่ 2.18 มัลติเพิลเฟรมทรานสมิตชั่น

MAC Database Instability คือ Switch ได้รับ MAC Address ของอุปกรณ์เดียวเข้ามาหลายทาง ทำให้ Switch เรียนรู้ MAC Address ที่ผิดพลาด ดังภาพที่ 2.19



ภาพที่ 2.19 แมคแอดเดรสอินสทะบิลลิที

2.5.2. สแปนนิ่งทรีโปรโตคอลคืออะไร Spanning-Tree Protocol (STP) เป็น Protocol ที่ใช้ป้องกันลูป (Loop) ใน Layer 2 โดยการทำงานคือจะ Blocking Port เพื่อไม่ให้รับส่งข้อมูลจนกว่าเส้นทางหลักจะมีปัญหา และยังช่วยเสริมให้มีเส้นทางสำรอง เช่น สมมุติว่าเรามีจุดหมายปลายทางอยู่จุดหนึ่งแล้วเส้นทางนี้เกิดมีปัญหาทำให้ระบบใช้งานไม่ได้เลย ก็ทำให้ระบบทั้งหมดมีปัญหาไปด้วย ตัว Spanning Tree มันก็จะมีระบบช่วยป้องกันไม่ให้ระบบหยุดการทำงาน ถ้าเส้นทางหนึ่งมีปัญหาก็สามารถไปใช้เส้นทางอื่นได้ Redundancy ของ Spanning Tree มันทำให้ระบบมีเสถียรภาพ เพราะใช้ตลอดเวลาก็ไม่ Down ถึงแม้เส้นทางใดเส้นทางหนึ่งใช้ไม่ได้ก็ตาม Spanning tree ก็จะมีเส้นทางขึ้นมาใช้แทนโดยรวมทำให้มีเสถียรภาพมากขึ้น หากกำลัง

สงสัยว่าทำไม Protocol ตัวนี้จึงถูกนำมาใช้งานกับอุปกรณ์ Layer 2 นั้นก็เพราะว่าการเกิด Loop นั้นส่วนใหญ่จะเกิดมาจากอุปกรณ์ Layer 2 อย่าง Switch ในส่วนของอุปกรณ์ Layer 3 จะมี TTL(Time to Live) ไว้สำหรับป้องกันการเกิดลูปอยู่แล้ว

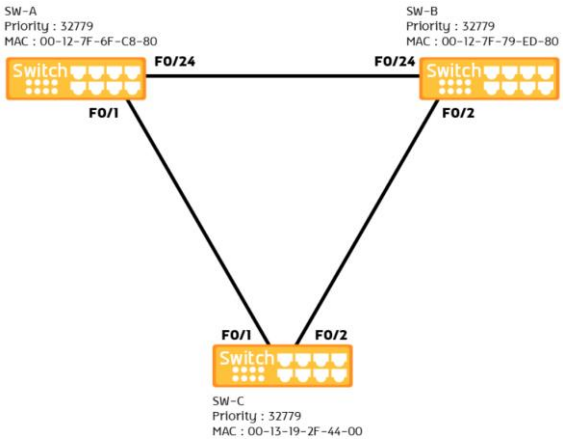
2.5.3. หลักการทำงานของ STPหลักการเลือกเส้นทางการส่งข้อมูลของ Spanning Tree มีขั้นตอนการพิจารณาดังต่อไปนี้

1. เลือก Root Bridge คือใน 1 Network จะมี Switch เพียง 1 ตัวที่เป็น Root Bridge พิจารณาจาก Switch ที่มี Bridge ID น้อยที่สุด (Bridge ID = Priority + MAC Address) และขาของ Bridge ID เป็น Designated Port

2. เลือก Root Port พิจารณาจาก Port ที่มีค่า Path Cost ไปหา Root Bridge ต่ำสุดถ้า Path Cost เท่ากันให้พิจารณาจาก Bridge ID โดย Switch จะมี Root Port ได้เพียง 1 ตัวเท่านั้น

3. เลือก Design Port ใน Link ระหว่าง Switch กับ Switch หรือที่เรียกว่า Segment ต้องมี 1 Designated Port โดยพิจารณาจาก Path Cost ไป Root Bridge ต่ำสุด หากเท่ากันให้พิจารณาจาก Bridge ID ต่ำสุด ถ้า Bridge ID เท่ากัน ให้พิจารณาจาก Port ID ต่ำสุด และ port ที่เหลือจาก Root Port และ Designated Port คือ Block Port

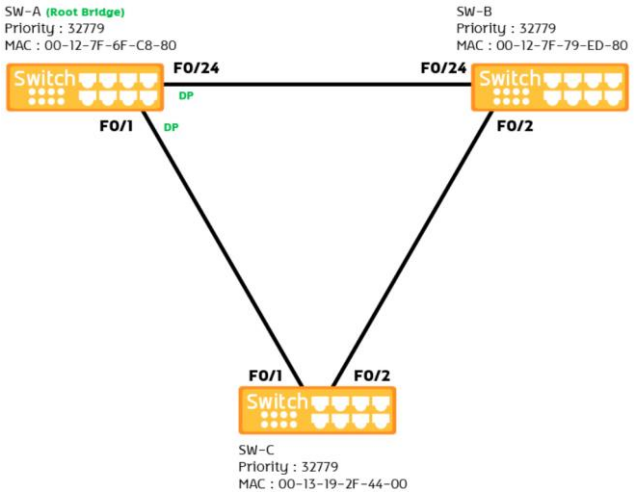
2.5.4 ตัวอย่างการคำนวณ STP ตัวอย่างการคำนวณหาค่ารูทบริทเพื่อป้องกันการเกิดลูปในอุปกรณ์เครือข่ายโดยใช้ STP นั้นสามารถคำนวณได้โดยวิธีการดังภาพที่ 2.20



ภาพที่ 2.20 การเชื่อมต่ออุปกรณ์สวิตช์เพื่อหารูทบริท

(ที่มา: https://www.pi-tech.biz/17253624/stp-spanning-tree-protocol)

จากภาพที่ 2.20 หลังจากทำการเชื่อมต่ออุปกรณ์สวิตช์เข้าได้กัน สวิตช์แต่ละตัวก็จะมีค่าไพรออริตี้และค่าแมคแอดเดรสไว้สำหรับการกำหนดค่ารูทบริทให้กับอุปกรณ์ การหารูทบริท (Root Bridge) พิจารณาจากบริทไอดี (Bridge ID) ที่ต่ำสุด คือ SW-A และกำหนด ดีไซน์เนสพอร์ต (Design Port) ตัวย่อ ดีพี (DP) ดังภาพที่ 2.21



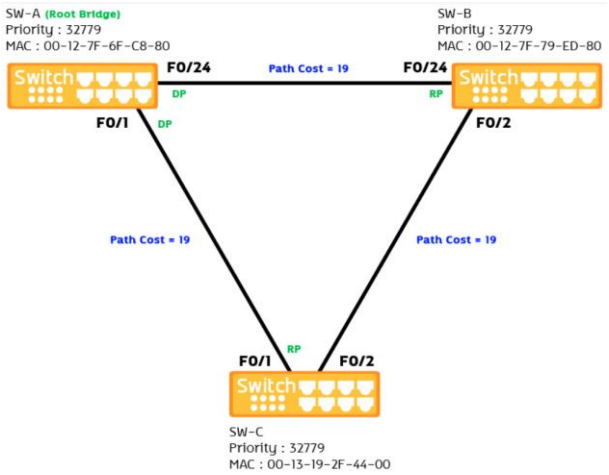
ภาพที่ 2.21 กำหนด ดีไซน์เนสพอร์ต (Design Port) ให้กับ SW-A

(ที่มา: https://www.pi-tech.biz/17253624/stp-spanning-tree-protocol)

จากภาพที่ 2.21 เป็นการหารูทบริท (Root Bridge) โดยพิจารณาจากบริทไอดี (Bridge ID) ที่ต่ำสุด คือ SW-A และกำหนด ดีไซน์เนสพอร์ต (Design Port) ตัวย่อ ดีพี (DP) ต่อไปจะเป็นการหา รูทพอร์ต (Root Port) โดยพิจารณาจาก พาทคอร์ด (Path Cost) ในที่นี้เป็นฟาสอีเธอร์เน็ต (FastEthernet) ซึ่งมีค่าคอร์อยู่ที่ 19 ให้พิจารณาแต่ละตัวดังนี้

- SW-C ไปยัง SW-A ผ่าน Fa 0/1 = 19,
- SW-C ไปยัง SW-A ผ่าน Fa 0/2 และ Fa 0/24 = 38,
- SW-B ไปยัง SW-A ผ่าน Fa 0/24 = 19,
- SW-B ไปยัง SW-A ผ่าน Fa 0/2 และFa 0/1 = 38

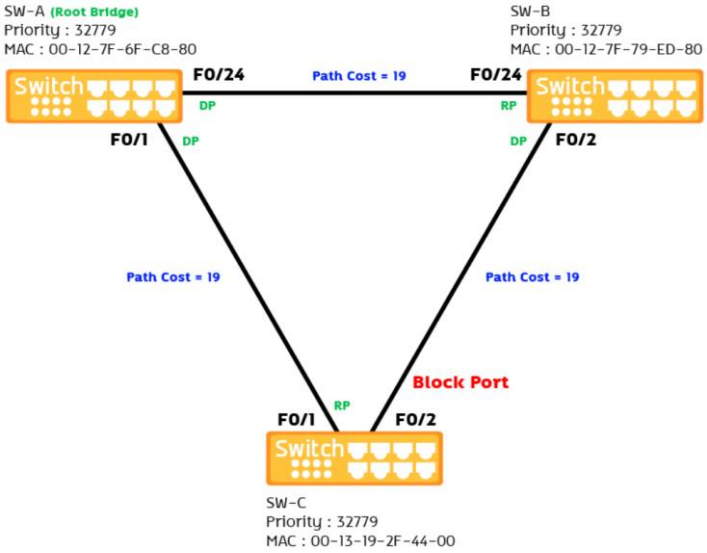
จะได้ Switch ที่มี รูทพอร์ต (Root Port) ที่ SW-B ที่ FastEthernet 0/24 และ SW-C ที่ FastEthernet 0/1 โดยการเลือกค่า รูทพอร์ต (Root Port) จะเลือกผลรวมของค่า พาทคอร์ด (Path Cost) ที่น้อยที่สุดได้ผลลัพธ์ดังภาพที่ 2.22



ภาพที่ 2.22 กำหนด รูทพอร์ต (Root Port) ให้กับ SW-C และ SW-B

(ที่มา: https://www.pi-tech.biz/17253624/stp-spanning-tree-protocol)

จากภาพที่ 2.22 หลังจากหารูทพอร์ต (Root Port) ให้กับอุปกรณ์เครือข่ายได้แล้ว ขั้นตอนต่อไปจะเป็นการหา ดีไซน์เนสพอร์ต (Designated Port) ในที่นี้จะมี พาทคอร์ด (Path Cost) เท่ากัน ให้พิจารณาจาก รูทบริทไอดี (Bridge ID) และผลลัพธ์ที่ได้คือ ดีไซน์เนสพอร์ต (Designated Port) อยู่ที่ SW-B ที่ FastEthernet 0/2 และ บล็อกพอร์ต (Block Port) คือ SW-C ที่ Fa 0/2 ดังภาพที่ 2.23

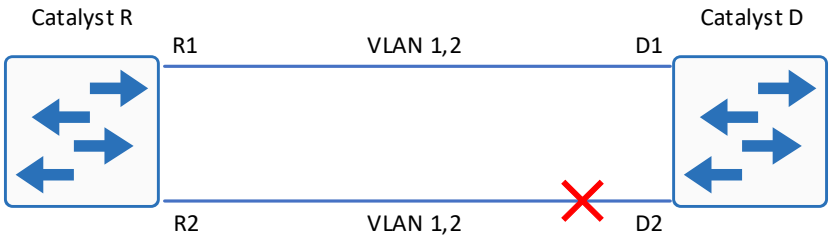


ภาพที่ 2.23 กำหนด รูทพอร์ต (Root Port) ให้กับ SW-C และ SW-B

(ที่มา: https://www.pi-tech.biz/17253624/stp-spanning-tree-protocol)

2.5.5. มัลติเพิลสแปนนิ่งทรีโปรโตคอล Multiple Spanning Tree Protocol (MSTP) และอัลกอริทึมให้ทั้งการเชื่อมต่อที่เรียบง่ายและเต็มรูปแบบที่กำหนดให้กับ Virtual LAN (VLAN) ใดๆ ที่ระบุทั่วทั้ง Bridged Local Area Network MSTP ใช้ BPDU เพื่อแลกเปลี่ยนข้อมูลระหว่างอุปกรณ์ที่เข้ากันได้กับ spanning-tree เพื่อป้องกันลูปในแต่ละ MSTI (Multiple Spanning Tree Instances) และใน CIST (Common and Internal Spanning Tree) โดยการเลือกเส้นทางที่ใช้งานอยู่และที่ถูกบล็อก สิ่งนี้ทำได้เช่นเดียวกับใน STP โดยไม่จำเป็นต้องเปิดใช้งานลิงก์สำรองด้วยตนเองและกำจัดอันตรายของบริดจ์ลูป นอกจากนี้ MSTP ยังอนุญาตให้เฟรม/แพ็กเก็ตที่กำหนดให้กับ VLAN ที่แตกต่างกันไปตามพาธที่แยกจากกัน โดยแต่ละพาธอิงตาม MSTI อิสระ ภายในภูมิภาค MST ที่ประกอบด้วย LAN และหรือบริดจ์ MST ภูมิภาคเหล่านี้และบริดจ์และ LAN อื่นๆ เชื่อมต่อเป็น Common Spanning Tree (CST) เดียว เนื่องจาก MSTP เปิดใช้งานการจัดกลุ่มและการแมป VLAN ลงในอินสแตนซ์ spanning tree ที่แตกต่างกัน จึงมีความจำเป็นต้องกำหนดกลุ่มหรือชุดของ VLAN ซึ่งทั้งหมดใช้ spanning tree เดียวกัน นี่คือสิ่งที่เรามารู้จักในชื่อ MSTI แต่ละอินสแตนซ์กำหนดโทโพโลยีการส่งต่อเดียวสำหรับชุด VLAN พิเศษ ในทางตรงกันข้าม เครือข่าย STP หรือ RSTP มีอินสแตนซ์แบบขยายเดี่ยวสำหรับเครือข่ายทั้งหมด ซึ่งประกอบด้วย VLAN ทั้งหมด

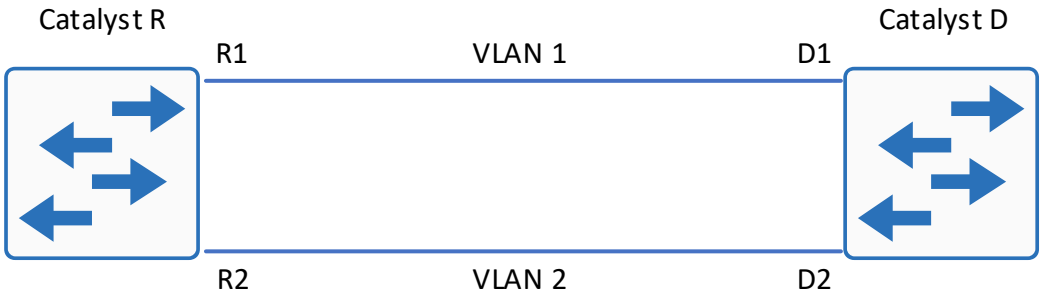
2.5.6. วีแลนโหลดบาลานซ์ระหว่างทรังค์โดยใช้สแปนนิ่งทรีโปรโตคอล (VLAN Load Balancing Between Trunks Using the Spanning-Tree Protocol) การกำหนดค่าที่แสดงในภาพที่ 2.24 ซึ่งสวิตช์สองตัวเชื่อมต่อโดยตรงผ่านทรังค์ (Trunk) เมื่อลิงก์ทั้งสองอุปกรณ์ใช้ สแปนนิ่งทรีอัลกอริทึม (Spanning Tree Algorithm) จะปิดใช้งานหนึ่งในลิงก์เพื่อหลีกเลี่ยงการลูปทั้งสองสวิตช์ หากหนึ่งในสองลิงก์ล้มเหลว ลิงก์ที่สองจะพร้อมสำหรับการส่งข้อมูลการจราจร (Traffic) ในเวลาต่อมา ดังภาพที่ 2.24



ภาพที่ 2.24 ข้อมูลการจราจรในสแปนนิ่งทรีอัลกอริทึม

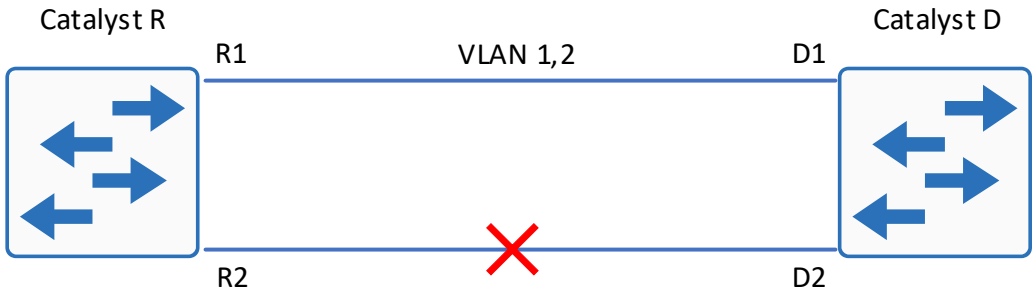
จากภาพที่ 2.24 มีการกำหนดทรังค์ (Trunks) สองตัวที่เชื่อมโยงกันระหว่าง Catalyst R และ Catalyst D สแปนนิ่งทรีจะเลือกพอร์ตการบล็อกลิงก์เดียว สำหรับ VLAN ทั้งหมดที่กำหนดผ่านทรังค์ ในกรณีนี้ Catalyst R คือ รูทบริท (Root Bridge) และ Catalyst D ตัดสินใจบล็อกพอร์ต D2 สำหรับทั้งวีแลนหนึ่ง (VLAN 1) และ วีแลนสอง (VLAN 2) ปัญหาหลักของการออกแบบนี้คือลิงก์ที่เชื่อมโยงกันหว่าง R2 และ D2

จะไม่สามารถใช้งานได้ ระหว่างสวิตช์ทั้งสอง หากต้องการใช้ประโยชน์จากทั้งสองลิงก์ สามารถเปลี่ยนการกำหนดค่าและอนุญาต VLAN 1 เฉพาะบนลิงก์ R1-D1 และ VLAN 2 เท่านั้นบนลิงก์ R2-D2 ดังภาพที่ 2.25



ภาพที่ 2.25 การกระจายรูทบริทวีแลนสำหรับการบาลานซ์ทราฟฟิก

จากภาพที่ 2.25 ผลลัพธ์ที่ได้คือทั้งสองลิงก์สามารถส่งข้อมูลทั้งวีแลนหนึ่งและวีแลนสองได้พร้อมกัน อย่างไรก็ตาม หากลิงก์ใดลิงก์หนึ่งล้มเหลว จะสูญเสียการเชื่อมต่อสำหรับวีแลนใดวีแลนหนึ่งโดยสมบูรณ์ แล้วสแปนนิ่งทรีจะทำการปรับการเชื่อมต่อใหม่ ให้วีแลนที่อยู่ในลิงก์ที่ล้มเหลวสามารถส่งข้อมูลผ่านลิงก์ที่ใช้งานได้ปกติ โดยผลลัพธ์ดังภาพที่ 2.26



ภาพที่ 2.26 การปรับทราฟฟิกวีแลนโดยสแปนนิ่งทรี

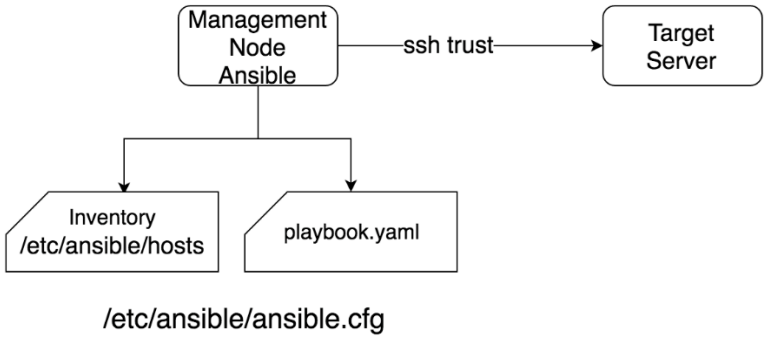
จากภาพที่ 2.26 ผลลัพธ์ที่ได้คือทั้งสองลิงก์สามารถส่งข้อมูลทั้งวีแลนหนึ่งและวีแลนสองปกติ อย่างไรก็ตาม หากลิงก์ที่มีการล้มเหลวมีสามารถใช้งานได้ตามปกติ สแปนนิ่งทรีจะทำการปรับการเชื่อมต่อให้อยู่ในรูปแบบเดิมดังภาพที่ 2.26

2.6 เอ็นซิเบิล Ansible

Ansible คือซอฟต์แวร์โอเพ่นซอร์ ที่ทำการจัดการการกำหนดค่าและมีเครื่องมือปรับใช้แอปพลิเคชัน ซึ่งการกำหนดค่าระบบ Ansible ถูกเขียนขึ้นโดย Michael DeHaan และได้มาจาก Red Hat ในปี 2015 Ansible นั้นไม่มีเอเจนต์ โดยเชื่อมต่อจากระยะไกลผ่าน SSH หรือ Windows Remote Management เพื่อทำงาน โดย Ansible เป็นซอฟต์แวร์การจัดการคอนฟิกูเรชันที่ขึ้นกับ Python โดยที่ทั้งโหนดควบคุมและเครื่องเป้าหมายต้องติดตั้ง Python และแพ็กเกจที่ขึ้นต่อกัน Ansible โมดูล Ansible Network เป็นส่วนขยายที่มีประโยชน์มากสำหรับระบบอัตโนมัติที่เรียบง่าย ทรงพลัง และไม่ต้องใช้เอเจนต์ให้กับผู้ดูแลระบบเครือข่าย การนำ Ansible มาใช้งานในระบบเครือข่ายนั้นมีจุดประสงค์เพื่อกำหนดค่าอุปกรณ์เครือข่าย โดยอุปกรณ์เครือข่ายที่สามารถนำมาใช้ได้นั้นจำเป็นต้องมีการซัพพอร์ตกับ Ansible จึงจะสามารถใช้งานได้ การทำงานของ Ansible จะประกอบด้วยไฟล์ที่เกี่ยวข้อง 2 ไฟล์คือไฟล์ inventory และ playbook

2.6.1. inventory ทำหน้าเก็บรายชื่อของเซิร์ฟเวอร์หรือไอพีของเซิร์ฟเวอร์ที่จะใช้สำหรับเป้าหมายสำหรับการติดตั้ง โดยค่าเริ่มต้นไม่มีการกำหนดในระบบก็จะใช้ inventory ที่อยู่ใน /etc/ansible/hosts สามารถที่จะแก้ไขค่านี้ได้โดยทำการแก้ไขไฟล์ /etc/ansible/ansible.cfg

2.6.2. playbook ไฟล์นี้จะเป็นตัวระบุว่าจะให้ Ansible ทำอะไรบ้างในแต่ละเซิร์ฟเวอร์การเขียนไฟล์จะใช้เป็นรูปแบบของ yaml โดยสามารถเขียนแบบที่เป็น static และ dynamic ก็ได้โดยถ้าหาเขียนแบบ dynamic โดยใช้ jinja2 template ในการทดสอบการใช้งานจะทดสอบการใช้งานผ่านทาง Vagrantfile โดยจะเป็นการสร้าง infrastructure ด้วย Vagrantfile ดังนี้ โดยสร้างบน provider virtualbox ดังภาพที่ 2.27



ภาพที่ 2.27 Ansible Configuration Management

2.7 งานวิจัยที่เกี่ยวข้อง

นายวิศรุต คุ้มเงิน (2561) การจัดสรรแบนด์วิดท์แบบละเอียดในเครือข่ายกำหนดด้วยซอฟต์แวร์ Fine-Grained Bandwidth Allocation in Software-Defined Networks ลักษณะงานวิจัย มีวัตถุประสงค์หลักเพื่อศึกษาการจัดสรรทรัพยากรแบนด์วิดท์แบบละเอียดภายในเครือข่ายกำหนดด้วยซอฟต์แวร์ตามความต้องการของผู้ใช้ (Fine-Grained Bandwidth Allocation in Software-Defined Networks) เพื่อให้เกิดการใช้งานแบนด์วิดท์ของ ช่องสื่อสารที่มีอยู่อย่างจำกัดให้เกิดประโยชน์และมีประสิทธิภาพสูงสุด เช่น กรณีที่เกิดภัยพิบัติ ช่องสื่อสารถูกทำลายและเหลือเพียงช่องสื่อสารเดียว จึงต้องเรียงลำดับความสำคัญของแต่ละแพ็กเก็ตโฟลว์ที่เข้ามาใช้ช่องสื่อสารเดียวกัน โดยแพ็กเก็ตโฟลว์ที่มีลำดับความสำคัญจะได้รับแบนด์วิดท์ มากกว่าแพ็กเก็ตโฟลว์ที่มีลำดับความสำคัญรองลงมาดำเนินงานเพื่อพัฒนาตัวจัดการทรัพยากรแบนด์วิดท์แบบละเอียดภายในเครือข่ายกำหนดด้วยซอฟต์แวร์ หรือ เอฟจีแบม (Fine-Grained Bandwidth Allocating Manager - FGBAM) ซึ่งมีลักษณะการทำงานแบบพร็อกซี่ (proxy) อยู่ระหว่างเอสดีเอ็นคอนโทรลเลอร์ (SDN controller) และอุปกรณ์เครือข่าย ตัวจัดการทรัพยากรแบนด์วิดท์เอฟจีแบมประกอบไปด้วยส่วนต่อประสานกับผู้ใช้ที่เป็นโปรแกรมประยุกต์เว็บ (Web application) และส่วนที่ใช้ดักจับ ข้อความรูปแบบโอเพนโฟลว์ (OpenFlow message) และจัดการฐานข้อมูลของโอเพนวีสวิตช์ (OpenvSwitch Database Management protocol) เพื่อแก้ไขข้อจำกัดและรักษาคุณสมบัติที่ดีของงานวิจัยที่เกี่ยวข้องไว้ ตัวจัดการทรัพยากรแบนด์วิดท์ที่ทำการพัฒนาขึ้นควรมีคุณสมบัติพึงประสงค์สี่ ประการ ได้แก่ สามารถจัดสรรแบนด์วิดท์ของแต่ละช่องสื่อสารแบบละเอียด (fine-grained allocation) เพื่อให้แพ็กเก็ตโฟลว์ที่แตกต่างกันได้รับแบนด์วิดท์จากช่องสื่อสารร่วมกัน ได้ ส่งผลให้ สามารถใช้ทรัพยากรแบนด์วิดท์ของช่องสื่อสารที่มีอยู่อย่างจากัดได้เกิดประโยชน์และมีประสิทธิภาพสูงสุด

Ankur Chowdhary, Vaibhav Hemant Dixit, Naveen Tiwari, Sukhwa Kyung, Dr. Dijiang Huang and Dr. Gail-Joon (2017) ของมหาวิทยาลัยรัฐแอริโซนา โดยใช้วิธีการ SDN ทำเกี่ยวกับ (Science DMZ: SDN based Secured Cloud Testbed) ลักษณะงานวิจัย เป็นการใช้ SDN ป้องกันการละเมิดความปลอดภัย มีการแฮ็คของ IRS ในปี 2559 มีรายละเอียด 100,000 FAFSA ผู้สมัครนักศึกษา หน่วยงานราชการ, องค์กร, การศึกษาสถาบันและภาคอื่นๆ กำลังเผชิญกับการเพิ่มขึ้นจำนวนภัยคุกคามความปลอดภัย จึงนำ SDN มาช่วยให้การจัดการแบบรวมศูนย์ของพื้นฐานเครือข่ายเมื่อเทียบกับเครือข่ายดั้งเดิม ความก้าวหน้าล่าสุดในการปรับใช้ testbeds ทดลองเช่น GENI อนุญาตเครือข่ายมหาวิทยาลัยที่จะเชื่อมต่อซึ่งกันและกัน GENI ยังได้แนะนำความเป็นไปได้ในการออกแบบโปรโตคอลเครือข่ายใหม่การจัดการเนื้อหาและเครือข่ายการปรับใช้บริการ อย่างไรก็ตามการทำงานกับ SDN มี จำกัด testbeds ความปลอดภัยที่ใช้สำหรับเครือข่ายมหาวิทยาลัยเราแสดงให้เห็นถึงชุดทดสอบ DMZ ที่ปลอดภัยสำหรับการจัดการการรับส่งข้อมูลเครือข่ายการระบุที่น่าสงสัยโจมตีเวกเตอร์และตอบโต้ที่จำเป็นเพื่อป้องกันการละเมิดความปลอดภัย Science DMZ ช่วยให้เราสามารถใช้

ประโยชน์ได้SDN สำหรับเหตุการณ์ด้านความปลอดภัยเหตุการณ์นโยบายการรับส่งข้อมูลเครือข่ายการจัดการและการจดจำรูปแบบการโจมตี วิทยาศาสตร์กรอบ DMZ จะตรวจจับและแก้ไข SDN ใดๆ โดยอัตโนมัติการละเมิดกฎการไหลตัวอย่างของ Science DMZ นำเสนอการจัดการและความปลอดภัยของเครือข่ายคลาวด์โดยใช้ SDN ขณะนี้เรามีใช้งานส่วนประกอบหลักเช่นไฟร์วอลล์ที่เปิดใช้งาน SDN และ Flow Checker Checker, honeypot และโหลด SDN balancer เครื่องวิเคราะห์ความปลอดภัย Science DMZ GUI ของเราอนุญาตผู้ใช้ระบบในการประสานและควบคุมปัญหาด้านความปลอดภัยในสภาพแวดล้อมคลาวด์ เราวางแผนที่จะขยายส่วนการวิเคราะห์บันทึกของระบบเพื่อระบุสถานการณ์การคุกคามขั้นสูงแบบถาวร (APT) และทำการวิเคราะห์พฤติกรรมของผู้โจมตี

ปิยพงษ์ เคนเหลื่อม (2018) คุณภาพการให้บริการสตรีมมิ่งที่มีการจัดความสำคัญบนเครือข่ายเอสดีเอ็น มหาวิทยาลัยธุรกิจบัณฑิตย์ ลักษณะงานวิจัย การใช้บริการระบบสตีมมิ่งภาพและเสียงโดยทั่วไปนั้นมีการแบ่งกลุ่มผู้ใช้งานตามความสำคัญเช่นกลุ่มผู้ใช้งานที่มี ความสำคัญหรือกลุ่มพรีเมียมกับกลุ่มผู้ใช้งานปกติ ซึ่งกลุ่มพรีเมียมนั้น ต้องมีการบริการที่ดีกว่า แบนด์วิดท์ถือเป็นทรัพยากรที่ต้องมีการบริหารให้มีประสิทธิภาพ แต่ด้วยความหลากหลายของอุปกรณ์ทำให้เกิดปัญหาไม่สามารถควบคุมจัดการโดยระบบเพียงระบบเดียวหรือมีความยุ่งยากในการจัดการ อย่างไรก็ตามด้วยการใช้งานเครือข่าย Software Defined Network หรือ SDN ซึ่งมีการควบคุมการทำงานด้วย OpenFlow การจัดระดับความสำคัญของกลุ่มผู้ใช้งานสามารถทำได้โดยง่าย ยืดหยุ่น และซับซ้อนกว่ามาก สามารถตอบสนองต่อความต้องการที่เปลี่ยนแปลงได้ ดังนั้นในงานวิจัยนี้จึงเสนอขั้นตอนการจัดการระดับ ความสำคัญ ของกลุ่มผู้ใช้งานเพื่อใช้บริการ สตรีมมิ่งบนเครือข่าย SDN เพื่อรับประกันแบนด์วิดท์ผ่านการควบคุมทาง North Bound ของ SDN ด้วยการเรียกใช้งาน web service ที่เป็น REST API เพื่อทำการการันตีแบนด์วิดท์ซึ่งการทดสอบจะแบ่งผู้ใช้งานออกเป็น 2 กลุ่มคือกลุ่มที่ให้ระดับความสำคัญและการันตีแบนด์วิดท์และกลุ่มผู้ใช้งานปกติที่ใช้งานทราฟฟิกจากแบนด์วิดท์ที่หลืออยู่ให้มากที่สุด

จากการศึกษาจะเห็นได้ว่า การจัดการระดับความสำคัญของผู้ใช้งานบน SDN นั้นทำได้ง่ายและมีความเป็นพลวัตสูงสามารถสร้างรูปแบบการใช้งานเครือข่ายได้ตามการประยุกต์ใช้ งานที่ต้องการ สำหรับการทดลองพบว่ากลุ่มผู้ใช้งานที่มีระดับ ความสำคัญสามารถทำงานได้ตามต้องการมีการส่งผ่านข้อมูลที่มี throughput ขั้นต่ำ ใกล้เคียงกับแบนด์วิดท์ที่ได้รับการรับประกันและมี jitter ที่อยู่ในเกณฑ์ที่ดีมาก ในขณะที่กลุ่มผู้ใช้งานทั่วไปที่ไม่ได้จัดระดับความสำคัญให้สามารถใช้งานได้ แต่แบนด์วิดท์จะถูกแบ่งปันไปให้กลุ่มผู้ใช้งานที่มีระดับความสำคัญกว่า ซึ่ง เป็นไปตามวัตถุประสงค์ที่ได้ออกแบบไว้ การศึกษาต่อไปอาจสร้างกลุ่มผู้ใช้งานที่มีความหลากหลายและมีระดับความสำคัญที่แตกต่างกันมากขึ้น รวมทั้งการศึกษาการใช้งานผสมผสานระหว่างสตรีมมิ่งและการใช้งานอื่นๆ