ซิพทราฟฟิกเจเนอเรเตอร์และอนาไลเซอร์

SIP TRAFFIC GENERATOR AND ANALYZER

วุฒินัย กาญจนศร
WUTTHINAI KANJANASORN

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์
บัณฑิตวิทยาลัย
สถาบันเทคโนโลยีพระจอมเกล้าธนบุรี
ปีการศึกษา 2550

ซิพทราฟฟิกเจเนอเรเตอร์และอนาไลเซอร์

SIP TRAFFIC GENERATOR AND ANALYZER

วุฒินัย กาญจนศร

WUTTHINAI KANJANASORN

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ. 2550

SIP Traffic Generator and Analyzer

WUTTHINAI KANJANASORN

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF ENGINEERING IN COMPUTER ENGINEERING

SCHOOL OF GRADUATE STUDIES

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2007

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใบรับรองวิทยานิพนธ์

หัวข้อวิทยานิพนธ์ ซิพทราฟฟิกเจเนอเรเตอร์และการวิเคราะห์ประสิทธิภาพ

SIP Traffic Generator and Performance Analysis

นักศึกษา นายวุฒินัย กาญจนศร

รหัสประจำตัว 47060810

ปริญญา วิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชา วิศวกรรมคอมพิวเตอร์

อาจารย์ที่ปรึกษาวิทยานิพนธ์ ผศ.ดร.สุรินทร์ กิตติธรกุล

คณะกรรมการสอบวิทยานิพนธ์		ลายมือชื่อ
ดร.วรวัฒน์	ลิ้มโภคา	
ผศ.ดร.ศักดิ์ชัย	ทิพย์จักษุรัตน์	
ผศ.ดร.เผ่าภัค	ศิริสุข	
ผศ.ธนา	หงษ์สุวรรณ	
ผศ.ดร.สุรินทร์	กิตติธรกุล	

วัน / เดือน/ ปี ที่สอบ 24 เมษายน 2550 เวลา 11.30-13.30 น.

สถานที่สอบ ณ อาคาร 12 ชั้น ชั้น 4 (ห้อง E12-404)

บัณฑิตวิทยาลัยรับรองแล้ว

(รศ.ดร.จารุวัตร เจริญสุข)

คณบดีบัณฑิตวิทยาลัย

วันที่.....6.....**เดือน**.....กรกฎาคม.....**พ.ศ**.....๒๕๕๐.....

หัวข้อวิทยานิพนธ์	ซิพทราฟฟิกเจเนอเรเตอร์และอนาไลเซอร์
นักศึกษา	นายวุฒินัย กาญจนศร
รหัสประจำตัว	47060810
ปริญญา	วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
พ.ศ.	2550
อาจารย์ผู้ควบคุมวิทยานิพนธ์	ผศ.ดร สุรินทร์ กิตติธรกุล

บทคัดย่อ

ซิพทราฟฟิกเจเนอเรเตอร์และอนาไลเซอร์เป็นโปรแกรมที่ใช้งานสำหรับทดสอบประสิทธิภาพของซิฟเซอร์เวอร์รวมทั้งเครือข่ายและอุปกรณ์ต่างๆที่เกี่ยวข้อง ซึ่งซอฟต์แวร์จะจำลองสถานการณ์ให้เหมือนกับกำลังมีการใช้งานจากผู้ใช้จำนวนมากๆได้ โดยที่มีความใกล้เคียงกับการใช้งานจริงหรือสามารถสร้างสถานการณ์ขึ้นเองได้ตามความต้องการ และมีการวิเคราะห์หรือแสดงผลที่เกิดขึ้นจากการทดลองได้ ทั้งนี้ซิพทราฟฟิกเจเนอเรเตอร์และอนาไลเซอร์ได้ทำการพัฒนาต่อมากจาก SIPp ที่เป็นโอเพ่นซอร์สแล้วทำการพอร์ทจากที่ทำงานบนลีนุกซ์มาเป็นทำงานบนวินโดวส์ ซึ่งยังได้เพิ่มความสามารถให้ทำงานได้ใกล้เคียงกับมาตรฐานของซิพมากขึ้น สามารถจำลองสถานการณ์เป็นผู้ใช้หลายๆคนได้ ซึ่งโปรแกรมสำหรับทดสอบอื่นๆนั้นยังไม่สามารถทำได้

Thesis Title	SIP Traffic Generator and Analyzer
Student	Mr. Wutthinai Kanjanasorn
Student ID	47060810
Degree	Master of Engineering
Program	Computer Engineering
Year	2007
Thesis Advisor	Asst.Prof.Dr Surin Kittitornkun

ABSTRACT

SIP Traffic Generator and Analyzer is used for testing SIP server, networks, devices and analyzing their performance. It can emulate almost all real situations. SIP Traffic Call Generator and Analyzer is developed on SIPp, which is an open source program which running on Linux system and ported to run on Windows XP. Multimedia SIP Call Generator and Analyzer has been improved to comply with SIP standard and emulate multiuser situations that other programs can not.

กิตติกรรมประกาศ

คุณความดีอันใดที่บังเกิดจากวิทยานิพนธ์ฉบับนี้ ขอมอบแด่บิดาและมารดาของผู้วิจัย ผู้ที่คอยห่วงใย เข้าใจและให้การสนับสนุนในการศึกษามาโดยตลอด

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงลงได้ด้วยดี โดยได้รับความกรุณาจาก ผศ.ดร สุรินทร์ กิตติธรกุล อาจารย์ที่ปรึกษา ที่ได้ช่วยเหลือในการให้คำแนะนำ ความรู้ทางทฤษฎีต่างๆที่ใช้ และชี้แนะแนวทางในการแก้ปัญหาต่างๆอย่างทุ่มเทรวมทั้งฝึกฝนผู้วิจัยให้มีความสามารถในการทำวิจัยและพัฒนาได้อย่างมีประสิทธิภาพ ผู้วิจัยรู้สึกซาบซึ้งในความอนุเคราะห์จากท่านและขอกราบขอบพระคุณเป็นอย่างสูง

ขอขอบคุณกรรมการสอบวิทยานิพนธ์ทุกท่านที่ได้กรุณาให้คำแนะนำในทุกๆเรื่อง ทั้งวิธีแก้ปัญหาที่เกิดขึ้นในการทำวิทยานิพนธ์และมุมมองในเชิงวิศวกรรมอื่นๆซึ่งช่วยให้ผู้วิจัยมีวิสัยทัศน์ที่กว้างไกลขึ้น

ขอขอบคุณบัณฑิตวิทยาลัย สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ที่ให้การสนับสนุนการทำวิทยานิพนธ์นี้

ขอบคุณพี่ๆเพื่อนๆและน้องๆนักศึกษาทุกคนในห้องวิจัย รวมทั้งเพื่อนๆหลายๆคนในอินเตอร์เน็ตที่ช่วยเหลือให้คำแนะนำต่างๆและให้กำลังใจแก่ผู้วิจัยตลอดมา

วุฒินัย กาญจนศร

สารบัญ

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญรูป

สารบัญรูป (ต่อ)

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

เนื่องจากระบบการสื่อสารในปัจจุบันได้พัฒนาขึ้นมาก จนเข้าสู้ในยุคที่สามารถสื่อสารได้โดยผ่านทั้งทางข้อความ, ภาพและเสียง โดยเฉพาะระบบการสื่อสารของโทรศัพท์เคลื่อนที่หรือมือถือที่ได้มีถูกพัฒนาอย่างมากจนกำลังจะเข้ามาสู่ยุคที่สามแล้ว 3G (Third Generation) ซึ่งมี SIP (Session Initiation Protocol) RFC3261 [1] เป็น IETF โปรโตคอลที่ทำงานอยู่บนชั้นแอพพลิเคชั่นเลเยอร์ซึ่งใช้สำหรับควบคุมการสร้าง, แก้ไขและทำลายเซสชั่นของการติดต่อสื่อสารของคู่สื่อสารนั้น ซึ่ง SIP นั้นทำให้ขอบเขตของการใช้งานของแอพพลิเคชั่นไม่ได้จำกัดอยู่เฉพาะสัญญาณเสียงเท่านั้น แต่ยังขยายขอบเขตให้สามารถใช้งานรับส่งมัลติมีเดียผ่านไอพีได้ (กลายเป็น Instant Messaging หรือ Presence) และในปี 2000 นั้น 3GPP (Third Generation Partnership Program) ได้เลือกใช้ SIP เพื่อเป็นโปรโตคอลเซสชั่นสำหรับเครือข่ายโมบายล์ไอพีในยุคสามจี (3GPP Release 5) [2], [3], [4]

ดังนั้นเมื่อ SIP เริ่มมีการใช้งานกันมากขึ้น จึงจำเป็นที่จะต้องมีเครื่องมือที่ใช้สำหรับทดสอบความสามารถหรือประสิทธิภาพในการทำงานของซิพเซิร์ฟเวอร์และเครือข่ายที่ใช้งานนั้นๆได้ เพื่อนำไปปรับปรุงการตั้งค่าพารามิเตอร์ต่างๆให้สามารถรองรับจำนวนผู้ใช้งานได้มากขึ้นและมีประสิทธิภาพในการใช้งานของแต่ละกลุ่มผู้ใช้มากขึ้น และเพื่อวิเคราะห์ถึงปัญหาที่อาจจะเกิดขึ้นในส่วนต่างๆได้ เช่น ปัญหาคอขวดทางด้านประสิทธิภาพของเครือข่ายที่จุดต่างๆ ซึ่งตัววัดและทดสอบประสิทธิภาพอื่นๆที่มีอยู่ในปัจจุบันนั้นมักไม่เป็นไปตามมาตรฐาน RFC3261 โดยส่วนใหญ่ไม่สามารถจำลองการใช้งานเป็นหลายๆผู้ใช้งานในขณะเดียวกันได้ และไม่สามารถที่จะทำการสร้างแพ็กเก็ตเพื่อใช้งานผ่านทางพร็อกซี่จากผู้ใช้คนหนึ่งไปยังอีกคนหนึ่งได้ รวมทั้งเครื่องมือนี้ส่วนใหญ่จะอยู่ในระบบปฏิบัติการที่เป็นยูนิกซ์อีกด้วย จึงควรที่จะมี SIP Traffic Generator and Analyzer ที่นำมาใช้บนระบบปฏิบัติการที่เป็นวินโดวส์บ้าง และมีความสามารถที่จะสร้างแพ็กเก็ตหรือสถานการณ์ขึ้นมาใหม่เองให้เป็นไปตามมาตรฐาน และมีความเหมือนหรือคล้ายคลึงกับการใช้งานจริงๆให้มากที่สุด

1.2 วัตถุประสงค์ของการศึกษา

1. ศึกษาการทำงานของ SIP โปรโตคอล
2. ศึกษาวิธีในการวิเคราะห์ประสิทธิภาพในการทำงานของเซิร์ฟเวอร์, ไคลเอนต์และเครือข่ายที่ใช้งาน
3. พัฒนาเครื่องมือที่ใช้ในการวิเคราะห์ประสิทธิภาพในการใช้งาน SIP โดยที่เครื่องมือจะต้องมีความยืดหยุ่นในการใช้งานให้สอดคล้องกับเซิร์ฟเวอร์ที่แตกต่างกันได้ และมีความสามารถในการจำลองสถานการณ์ให้เหมือนหรือใกล้เคียงกันกับใช้งานจริงให้ได้มากที่สุด
4. เพื่อเป็นเครื่องมือที่สามารถนำไปทดสอบได้บนเครื่องที่เป็นวินโดวส์ซึ่งเป็นระบบปฏิบัติการที่นิยมมากที่สุด คือเพื่อความสะดวกในการนำไปใช้งาน
5. เพื่อให้สามารถวิเคราะห์ถึงประสิทธิภาพในการใช้งานให้ใกล้เคียงกับสถานการ์จริงได้ถูกต้องหรือใกล้เคียงมากที่สุด

1.3 สมมุติฐานของการศึกษา

เพื่อให้การพัฒนาซอฟต์แวร์ SIP Traffic Generator and Analyzer เป็นไปอย่างถูกต้องจึงต้องทำการศึกษารูปแบบการสื่อสารทั้งหมดของ SIP โปรโตคอล เพื่อให้ซอฟต์แวร์สามารถแสดงผลลัพธ์ในการทดสอบได้อย่างถูกต้อง ไม่ว่าจะเป็นสัญญาน INVITE, BYE, SUBSCRIBE, NOTIFY และอื่นๆ ซึ่งทั้งนี้ได้ทำการศึกษาตาม rfc3261 ซึ่งเป็นมาตรฐานของ SIP โปรโตคอล รวมทั้งยังต้องศึกษาการทำงานของซิพเซิรฟเวอร์จากผู้ผลิตหลายเจ้า เพื่อสังเกตถึงการทำงานที่แตกต่างกันของแต่ละผู้ผลิต เพื่อให้ซอฟต์แวร์ที่ได้พัฒนาขึ้นมาใหม่นี้มีความสามารถและมีความยืดหยุ่นให้สามารถนำไปใช้กับของผู้ผลิตซิพเซิรฟเวอร์ใดๆก็ได้ และยังต้องศึกษาถึงตัวทดสอบและวิเคราะห์ประสิทธิภาพของผู้อื่นเพื่อสร้างความแตกต่างและถมเติมซึ่งที่ขาดหายไปหรือข้อบกพร่องของผู้ผลิตให้ของเรามีความสามารถมากขึ้น

1.4 ทฤษฎีหรือแนวความคิดที่ใช้ในการวิจัย

วิทยานิพนธ์นี้ได้แนวความคิดมาจากการที่จะสร้างแบบทดสอบหรือสถานการณ์ยังไงให้มีความยืดหยุ่นเพื่อที่จะสามารถนำไปใช้ในการทดสอบซิพเซิรฟเวอร์ต่างๆกันได้อย่างมีประสิทธิภาพ ซึ่งในที่นี้เราได้กำหนดให้ผู้ใช้หรือผู้ทดสอบสามารถกำหนดสถานการณ์หรือ Scenario ได้จากการเขียน XML และเพื่อการทดสอบประสิทธิภาพของระบบ SIP นั้นๆ ตัวซอฟต์แวร์หรือ SIP Traffic Generator and Analyzer จึงต้องสามารถจำลองให้เหมือนกับมีผู้เข้ามาใช้งานระบบเป็นจำนวนมากๆได้ทีละหลาย user ที่ต่างกันพร้อมๆกัน และสามารถนำไปใช้งานในระบบปฏิบัติการที่เป็นวินโดวส์ได้

1.5 ขอบเขตของงานวิจัย

วิทยานิพนธ์นี้ได้ศึกษาถึงวิธีการในการวัดประสิทธิภาพ SIP server และ network โดยได้ทำการพัฒนาซอฟต์แวร์เพื่อใช้ในการนี้ด้วย ทั้งนี้ยังได้อธิบายถึงหลักการทำงานของ SIP โปรโตคอล และวิธีในการวัดประเมินประสิทธภาพของ SIP server และ network ด้วยซอฟต์แวร์ที่ได้พัฒนาขึ้น รวมทั้งได้ทำการวัดประสิทธิภาพของ SIP server ต่างๆ เช่น VOCAL และ partysip ซึ่งสุดท้ายก็จะได้ผลลัพธ์ออกมาในรูปของตัวเลขวัดผล อาทิเช่น response delay, interarival time, offered load, blocking probability, ErlangC และอื่นๆ ซึ่งในการพัฒนาซอฟต์แวร์นั้นจะพยายามอ้างอิงจาก rfc3261 หากเกิดความขัดแย้งขึ้นในการทดลองกับ SIP server ที่มีการทำงานที่แตกต่างกัน

1.6 ขั้นตอนการศึกษา

1. ศึกษาหลักการทำงานของ SIP โปรโตคอล
2. นำระบบ SIP ขึ้นมาใช้งานจริง เพื่อศึกษา
3. ศึกษาวิธีการต่างๆเพื่อที่จะวัดประสิทธิภาพในการใช้งานของ SIP server และ network ให้ได้ใกล้เคียงกันกับการใช้งานจริงมากที่สุด
4. ศึกษา tester และ analyzer ที่มีอยู่เพื่อนำมาหาข้อบกพร่องเพื่อทำการปรับปรุงแก้ไข
5. พัฒนาซอฟต์แวร์ SIP Traffic Generator and Analyzer
6. ทำการทดลองซอฟต์แวร์ที่ได้พัฒนาขึ้นโดยนำไปวัดประสิทธิภาพกับระบบ SIP ที่ได้นำมาใช้งานไรแต่ละระบบต่างๆกัน ว่าซอฟต์แวร์นั้นสามารถทำงานได้ภายใต้สภาวะหรือเงื่อนไขที่แตกต่างกันของ SIP แต่ละระบบ และทำการพัฒนาแก้ไขปรับปรุงต่อเพื่อกลับมาทดลองอีกครั้ง
7. สรุปและวิเคราะห์ผลการทำงานของซอฟต์แวร์

1.7 รายละเอียดของวิทยานิพนธ์

วิทยานิพนธ์ฉบับนี้ได้นำเสนอถึงวิธีการในการวัดประสิทธิภาพของระบบซิพทั้งระบบหรือบางส่วนของระบบ รวมทั้งซอฟต์แวร์ที่ได้ทำการพัฒนาขึ้นเพื่อใช้การวัดประเมินประสิทธิภาพ โดยรายละเอียดต่างๆภายในวิทยานิพนธ์นี้ได้จัดแบ่งในส่วนเนื้อหาออกเป็น 5 บท ซึ่งแต่ละบทมีหัวข้อและเนื้อหาดังต่อไปนี้

บทที่ 1 "บทนำ" อธิบายถึง ความเป็นมาและความสำคัญของปัญหา, วัตถุประสงค์ของการศึกษา,สมมุติฐานของการศึกษา, ทฤษฎีหรือแนวความคิดที่ใช้ในการวิจัย, ขอบเขตของงานวิจัยและขั้นตอนการศึกษา รวมไปถึงรายละเอียดเนื้อหาโดยสรุปของแต่ละบท

บทที่ 2 "SIP overview" อธิบายถึงหลักการทำงานของโปรโตคอลซิพโดยอ้างอิงตามมาตรฐาน rfc 3261 และมาตรฐานอื่นๆที่เกี่ยวข้อง ตั้งแต่ตัวอย่างการใช้งานง่ายๆ, SIP client and server, SIP request message และ SIP response message

บทที่ 3 "การออกแบบและพัฒนา" นำเสนอเครื่องมือหรืองานที่เกี่ยวข้อง, คุณสมบัติของ SIP Traffic Generator and Analyzer, การพัฒนาซอฟต์แวร์และลักษณะในการทำงาน

บทที่ 4 "การทดลองและผลการทดลอง" นำเสนอถึงวิธีการที่จะนำซอฟต์แวร์มาทดสอบว่าทำการทดสอบอย่างไร ซึ่งได้ทดสอบความถูกต้องโดยการนำผลลัพธ์ไปเปรียบเทียบกับ SIPp เดิมก่อนการแก้ไขเพิ่มเติม และยังได้ทดลองนำไปทดสอบกับ VOCAL และ Partysip ซึ่งเป็นซิพเซิร์ฟเวอร์และแสดงผลการทดลองใช้งานของซอฟต์แวร์โดยการนำซอฟต์แวร์ไปวัดประสิทธิภาพของเซิร์ฟเวอร์ทั้งสอง

บทที่ 5 "สรุป" เป็นการสรุปและวิจารณ์ผลที่ได้จากการวิเคราะห์และการทดลอง พร้อมทั้งปัญหาที่เกิดขึ้น ตลอดจนข้อเสนอแนะสำหรับนำงานวิจัยในวิทยานิพนธ์นี้ไปพัฒนาใช้ประโยชน์ต่อไป

บทที่ 2
ซิพโปรโตคอล

2.1 บทนำ

มีแอพพลิเคชั่นสำหรับอินเตอร์เน็ทที่ต้องการการสร้างหรือควมคุมจัดการเซสชั่น ที่ซึ่งเซสชั่นนั้นๆถูกใช้ในการแลกเปลี่ยนข้อมูลระหว่างคู่สื่อสารนั้นๆ การสร้างแอพพลิเคชั่นนั้นมีความสลับซับซ้อนที่เกิดจากคู่สื่อสารนั่นเอง คือ ผู้ใช้มักจะเคลื่อนที่ไปมาซึ่งทำให้แอดเดรสของผู้ใช้อาจมีหลากหลาย และพวกเขายังอาจจะสื่อสารกันด้วยหลายวิธีการที่แตกต่างกัน โปรโตคอลจำนวนมากจึงถูกสร้างขึ้นเพื่อรองรับกับรูปแบบการใช้งานที่หลากหลายของเรียลไทม์มัลติมีเดีย ยกตัวอย่างเช่น เสียง, วีดีโอ หรือข้อความ (SIP messages) นั้นทำเพื่อให้โปรโตคอลเหล่านี้สามารถทำงานได้โดยมันจะทำหน้าที่ในการค้นหาแอดเดรสของผู้ใช้หรือผู้สื่อสารทั้งสองฝ่าย และเป็นตัวอธิบายถึงสักษณะของเซสชั่นที่จะเกิดขึ้นได้ระหว่างผู้ใช้หรือผู้สื่อสาร เพื่อระบุถึงเซสชั่นที่จะใช้ให้เหมาะสมและเพื่อการทำงานอื่นๆ ซิพนั้นได้สร้างรูปร่างของโฮสต์ของเครือข่ายซิพ (proxy server) ที่ซึ่ง User Agent สามารถส่ง Registrations, Invitations และอื่นๆ ซิพเป็นเครื่องมือที่ว่องไวและเป็นการง่ายสำหรับสร้าง, แก้ไข และตัดเซสชั่นที่ซึ่งทำงานได้อย่างอิสระได้ไม่ขึ้นกับโปรโตคอลในระดับชั้นทรานสปอร์ตและประเภทของเซสชั่นที่ซึ่งจะถูกสร้างขึ้น

ซิพเป็นโปรโตคอลควบคุมในระดับชั้นแอพพลิเคชั่นที่ซึ่งสามารถสร้าง,แก้ไขและตัดเซสชั่นมัลติมีเดียอย่างเช่นการเรียกสายโทรศัพท์ผ่านทางอินเตอร์เน็ต สามารถ INVITE ผู้อื่นเพื่อเข้ามาร่วมยังเซสชั่นที่มีอยู่ได้ ยกตัวอย่างเช่นการประชุมสาย (conference) คือมีเดียสามารถถูกผนวกเข้ามายังเซสชั่นที่มีอยู่ได้ (และสามารถนำออกมาได้) ซิพรองรับการจับคู่ระหว่างชื่อและการบริการโอนสายที่ซึ่งรองรับความเป็นโมบิลิตี้ของบุคคลได้ ผู้ใช้สามารถรักษาความเป็นเอกลักษณ์ส่วนบุคคลได้โดยปราศจากสถานที่เครือข่าย

SIP รองรับ 5 หน้าที่ด้วยกันเพื่อสร้างหรือตัดการสื่อสารมัลติมีเดียคือ [5]

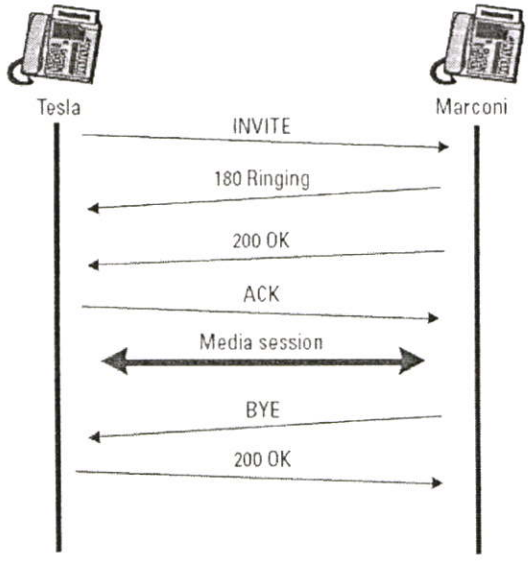
1. User location : กำหนดระบบปลายทางที่จะถูกใช้สำหรับสื่อสาร
2. User availability : กำหนดความต้องการของเครื่องที่เพื่อให้เกิดการสื่อสาร
3. User capabilities : กำหนดมีเดียและค่าตัวแปรของมีเดียที่จะถูกใช้
4. Session setup : "ringing" ก่อตั้งตัวแปรของเซสชั่นที่คู่สื่อสารและเครื่องมือที่ใช้
5. Session management : ประกอบด้วยการโอนถ่ายข้อมูลและการตัดการสื่อสาร, แก้ไข ตัวแปรของเซสชั่นและช่วยให้เกิดการให้บริการ

ซิพไม่ได้ถูกเพิ่มเข้าไปในระบบการสื่อสาร ซิพเป็นเพียงส่วนประกอบส่วนหนึ่งซึ่งสามารถถูกใช้ร่วมกับโปรโตคอล IETF เพื่อสร้างรูปแบบมัลติมีเดียให้สมบูรณ์ ซึ่งโดยทั่วไปแล้ว

รูปแบบเหล่านี้จะประกอบไปด้วยโปรโตคอลอย่างเช่น Real-time Transport Protocol (RTP) (RFC1889 [6]) สำหรับส่งถ่ายข้อมูลแบบเรียลไทม์และเตรียม QoS feedbask, Real-Time streaming protocol (RTSP) (RFC2326 [7]) เพื่อควบคุมการส่งมีเดียสตรีมมิ่ง, Media Gateway Control Protocol (MEGACO) (RFC3015 [8]) สำหรับควบคุมเกทเวย์ที่ Public Switched Telephone Network (PSTN) และ Session Description Protocol (SDP) (RFC2327 [9]) เพื่อบรรยายมิลติมีเดียเซสชั่น ดังนั้นซิพจึงควรจะถูกใช้ร่วมกันกับโปรโตคอลอื่นๆเพื่อให้เกิดการบริการที่สมบูรณ์แก่ผู้ใช้ อย่างไรก็ตามฟังก์ชั่นการทำงานหลักและการทำงานของซิพไม่ได้ขึ้นกับโปรโตคอลเหล่านี้

2.1.1 ตัวอย่างง่ายๆของการสร้างเซสชั่น

รูปที่ 2.1 แสดงซิพเมสเสจที่แลกเปลี่ยนกันระหว่างอุปกรณ์ซิพ 2 ตัว ซึ่งอุปกรณ์ทั้งสองตัวนั้นอาจจะเป็นโทรศัพท์ซิพ, ปาล์มหรือโทรศัพท์เคลื่อนที่ ซึ่งได้สมมุติว่ามันทั้งคู่เชื่อมต่อกับเครือข่ายไอพีและแต่ละตัวรู้ไอพีแอดเดรสซึ่งกันและกัน



รูปที่ 2.1 ตัวอย่างการสร้างเซสชั่นอย่างง่ายๆ

Tesla นั้นเริ่มที่จะส่งเมสเสจโดยการส่ง INVITE ไปยัง Marconi ซึ่ง INVITE จะประกอบไปด้วยรายละเอียดของประเภทของเซสชั่นที่ต้องการ ซึ่งอาจจะเป็นเสียง, มัลติมีเดียอย่างวีดีโอคอนเฟอร์เรนซ์หรือเกมส์

INVITE ประกอบด้วยฟิลด์ต่างๆดังนี้คือ

```
INVITE sip:marconi@radio.org SIP/2.0
Via: SIP/2.0/UDP lab.high-voltage.org:5060;branch=z9hG4bKfw19b
Max-Forwards: 70
To: G. Marconi <sip:Marconi@radio.org>
```

```
From: Nikola Tesla <sip:n.tesla@high-voltage.org>;tag=76341
Call-ID: 123456789@lab.high-voltage.org
CSeq: 1 INVITE
Subject: About That Power Outage...
Contact: <sip:n.tesla@lab.high-voltage.org>
Content-Type: application/sdp
Content-Length: 158
v=0
o=Tesla 2890844526 2890844526 IN IP4 lab.high-voltage.org
s=Phone Call
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

INVITE นี้เป็นตัวอย่างของ SIP request message มีด้วยกัน 5 SIP request ต่างๆกันที่ระบุใน RFC3261 และอื่นๆใน RFCs เพิ่มเติม มีเมสเสจต่อไปในรูปที่ 2.1 คือ 180 RINGING ที่ถูกส่งเป็นการตอบรับของ INVITE นี้เพื่อบอกว่า Marconi ได้รับ INVITE และได้ทำการเตือน (มีเสียงเรียกสาย) แล้ว

180 RINGING response มีโครงสร้างดังนี้

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP lab.high-voltage.org:5060;branch=z9hG4bKfw19b
;received=100.101.102.103
To: G. Marconi <sip:marconi@radio.org>;tag=a53e42
From: Nikola Tesla <sip:n.tesla@high-voltage.org>>;tag=76341
Call-ID: 123456789@lab.high-voltage.org
CSeq: 1 INVITE
Contact: <sip:marconi@tower.radio.org>
Content-Length: 0
```

เมื่อ Marconi ตัดสินใจที่จะรับสายแล้ว 200 OK response จะถูกส่ง เพื่อบอกว่าชนิดของมีเดียเซสชั่นตามที่ผู้เรียกเสนอมานั้นเป็นที่ยอมรับ

```
SIP/2.0 200 OK
```

```
Via: SIP/2.0/UDP lab.high-voltage.org:5060;branch=z9hG4bKfw19b
;received=100.101.102.103
To: G. Marconi <sip:marconi@radio.org>;tag=a53e42
From: Nikola Tesla <sip:n.tesla@high-voltage.org>;tag=76341
Call-ID: 123456789@lab.high-voltage.org
CSeq: 1 INVITE
Contact: <sip:marconi@tower.radio.org>
Content-Type: application/sdp
Content-Length: 155
v=0
o=Marconi 2890844528 2890844528 IN IP4 tower.radio.org
s=Phone Call
c=IN IP4 200.201.202.203
t=0 0
m=audio 60000 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

ขั้นสุดท้ายเพื่อเป็นการอนุญาติสร้างมีเดียเซสชั่นสามารถถูกสร้างขึ้นได้ภายใต้โปรโตคอลอื่นๆ ในตัวอย่างนี้คือ RTP

```
ACK sip:marconi@tower.radio.org SIP/2.0
Via: SIP/2.0/UDP lab.high-voltage.org:5060;branch=z9hG4bK321g
Max-Forwards: 70
To: G. Marconi <sip:marconi@radio.org>;tag=a53e42
From: Nikola Tesla <sip:n.tesla@high-voltage.org>;tag=76341
Call-ID: 123456789@lab.high-voltage.org
CSeq: 1 ACK
Content-Length: 0
```

ในรูปที่ 2.1 นั้น BYE request ถูกส่งโดย Marconi เพื่อเป็นการตัดเซสชั่น

```
BYE sip:n.tesla@lab.high-voltage.org SIP/2.0
Via: SIP/2.0/UDP tower.radio.org:5060;branch=z9hG4bK392kf
Max-Forwards: 70
To: Nikola Tesla <sip:n.tesla@high-voltage.org>;tag=76341
From: G. Marconi <sip:marconi@radio.org>;tag=a53e42
```

```
Call-ID: 123456789@lab.high-voltage.org
CSeq: 1 BYE
Content-Length: 0
```

และการยืนยันหรือการตอบรับของ BYE คือ 200 OK

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP tower.radio.org:5060;branch=z9hG4bK392kf
;received=200.201.202.203
To: Nikola Tesla <sip:n.tesla@high-voltage.org>;tag=76341
From: G. Marconi <sip:marconi@radio.org>;tag=a53e42
Call-ID: 123456789@lab.high-voltage.org
CSeq: 1 BYE
Content-Length: 0
```

2.1.2 SIP Call ด้วย Proxy server

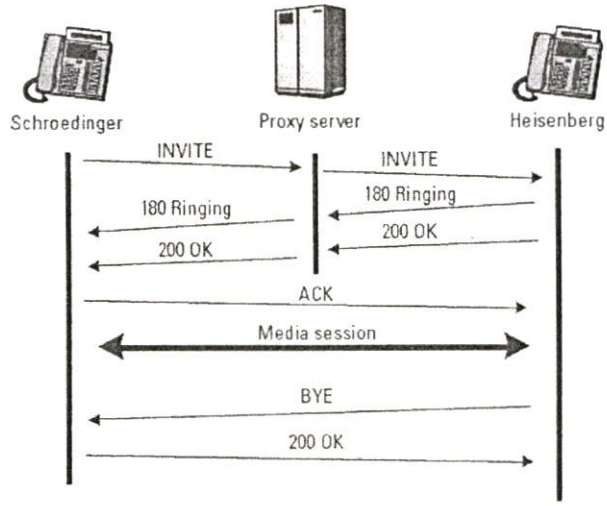
ซิพเมสเสจที่แลกเปลี่ยนกันในรูปที่ 2.1 นั้น Tesla รู้ถึงไอพีแอดเดรสของ Marconi ทำให้สามารถส่ง INVITE โดยใช้แอดเดรสนั้นๆได้เลย ซึ่งนี่ไม่ใช่กรณีปกติที่เกิดขึ้น คือ ไอพีแอดเดรสนั้นไม่สามารถถูกใช้เหมือนกันกับหมายเลขโทรศัพท์ได้ เหตุผลหนึ่งคือมันมักจะถูกใช้แบบ ไดนามิก อย่างเช่น เมื่อคอมพิวเตอร์ได้ทำการต่อสายไปยัง ISP นั้น ไอพีแอดเดรสที่ได้รับโดย DHCP จากแอดเดรสที่ยังมีเหลืออยู่ ก็คือในช่วงระยะเวลาหนึ่งๆนั้นไอพีแอดเดรสนั้นจะไม่มีการเปลี่ยนแปลงแต่ในการต่อสายแต่ละครั้งมันก็จะเกิดการเปลี่ยนแปลงได้ หรือแม้กระทั่งในกรณีที่ในขณะที่ใช้ไอพีแอดเดรสที่ที่ทำงานหรือที่บ้าน ก็ทำให้มีหลายไอพีแอดเดรสได้เช่นกัน ซึ่งในความเป็นจริงแล้วก็ยังมีอินเตอร์เน็ตโปรโตคอลที่ทำงานเกี่ยวกับอีเมล์คือ SMTP ที่โฮสต์หรือระบบจะเป็นอิสระจากไอพีแอดเดรสโดยการใช้อีเมล์แอดเดรสแทน

นอกจากนี้การที่ใช้เพียงไอพีแอดเดรสจะเป็นการกำหนดจุดสุดท้ายที่เป็นตัวอุปกรณ์เท่านั้น แต่การสื่อสารโดยทั่วไปนั้นจะเป็นแบบผู้ใช้ถึงผู้ใช้ไม่ใช่อุปกรณ์ถึงอุปกรณ์ดังนั้นการใช้ระบบแอดเดรสที่มีประสิทธิภาพจะทำให้ผู้ใช้สามารถเรียกสายไปยังผู้ใช้อีกคนได้โดยที่ไม่ต้องคำนึงถึงว่าผู้ใช้อีกคนใช้อุปกรณ์อันไหนอยู่ไหนกรณีที่มีหลายอุปกรณ์

ซิพใช้แอดเดรสในลักษณะที่คล้ายกับอีเมล์ซึ่งวิธีการใช้แอดเดรสแบบนี้ก็เป็นส่วนนึงของตระกูลอินเตอร์เน็ตแอดเดรสที่เรียกกันว่า URIs จุดสำคัญคือ SIP URI เป็นชื่อที่จะถูกแปลงเป็นไอพีแดแดรสโดยการใช้ SIP proxy server และ DNS ในขณะการเรียกสายแต่ละครั้ง รูปที่ 2.2 แสดงตัวอย่างที่มีการใช้งานส่วนมากของซิพที่มีการใช้ซิพเซิรฟเวอร์ที่เรียกว่า “proxy server” ในตัวอย่างนี้ผู้เรียก Schroedinger เรียก Heidenberg ผ่าน SIP proxy server ซึ่ง SIP proxy server จะทำงานคล้ายกันกับพร็อกซี่ของ HTTP หรือโปรโตคอลอื่นๆ SIP proxy server จะไม่สร้างหรือตัด

เซสชั่นแต่มันจะอยู่ระหว่างการแลกเปลี่ยนซิพเมสเสจคอยรับและส่งต่อพวกมัน ตัวอย่างนี้จะแสดงเพียงแต่พร็อกซี่เดียวแต่ในความเป็นจริงหลายๆพร็อกซี่ก็ได้

ซิพนั้นมี URI อยู่ 2 ประเภทก็คือ URI ที่ของตัวผู้ใช้กับ URI ของตัวอุปกรณ์หรือที่ปลายทาง ซึ่ง URI ของผู้ใช้จะเป็นที่รู้จักกันในชื่อ Address of record (AOR) ซึ่งการรีเควสที่ถูกส่งมายัง AOR นี้ต้องการการค้นหาในฐานข้อมูลและการบริการ และความสามารถที่จะสามารถทำให้ถูกส่งไปยังหลายๆอุปกรณ์ได้ ส่วน device URI จะถูกใช้เป็นฟิลด์ Contact และไม่ต้องการการค้นหาในฐานข้อมูลซึ่ง AOR URI นั้นจะถูกใช้ในเฮดเดอร์ฟิลด์ To และ From เพื่อเป็นการค้นหาให้ถึงตัวบุคคลและเหมาะที่จะเก็บ URI นี้เป็นสมุดแอดเดรส ส่วน device URI มักจะถูกใช้ในเฮดเดอร์ฟิลด์ Contact โดยมักจะถูกใช้ในช่วงระยะเวลาสั้นๆ



รูปที่ 2.2 ตัวอย่างการเรียกสายที่ใช้ proxy server

```
INVITE sip:werner.heisenberg@munich.de SIP/2.0
Via: SIP/2.0/UDP 100.101.102.103:5060;branch=z9hG4bKmp17a
Max-Forwards: 70
To: Heisenberg <sip:werner.heisenberg@munich.de>
From: E. Schroedinger <sip:schroed5244@aol.com>;tag=42
Call-ID: 10@100.101.102.103
CSeq: 1 INVITE
Subject: Where are you exactly?
Contact: <sip:schroed5244@pc33.aol.com>
Content-Type: application/sdp
Content-Length: 159
v=0
o=schroed5244 2890844526 2890844526 IN IP4 100.101.102.103
```

```
s=Phone Call
t=0 0
c=IN IP4 100.101.102.103
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

พร็อกซี่จะมอง SIP URI ใน Request-URI (sip:Werner.heisenberg@munich.de) ในฐานข้อมูลแล้วหาที่อยู่ของ Heisenberg เมื่อพร็อกซี่ได้รับ INVITE request ก็จะทำการส่งต่อ INVITE นี้ไปยัง Heisenberg โดยทำการเพิ่มฟิลด์ Via เข้าไปเพื่อบ่งบอกว่ารีเควสนี้ได้ผ่านพร็อกซี่มา

```
INVITE sip:werner.heisenberg@200.201.202.203 SIP/2.0
Via: SIP/2.0/UDP proxy.munich.de:5060;branch=z9hG4bK83842.1
Via: SIP/2.0/UDP 100.101.102.103:5060;branch=z9hG4bKmp17a
Max-Forwards: 69
To: Heisenberg <sip:werner.heisenberg@munich.de>
From: E. Schroedinger <sip:schroed5244@aol.com>;tag=42
Call-ID: 10@100.101.102.103
CSeq: 1 INVITE
Contact: <sip:schroed5244@pc33.aol.com>
Content-Type: application/sdp
Content-Length: 159
v=0
o=schroed5244 2890844526 2890844526 IN IP4 100.101.102.103
s=Phone Call
c=IN IP4 100.101.102.103
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

โดยหลังจากที่ Heisengerg ได้รับ INVITE นี้ก็จะทราบว่าเมสเสจนี้ได้ผ่านมาทางพร็อกซี่ จากนั้น Heisenberg ก็จะสร้าง 180 Ringing และส่งกลับผ่านไปทางพร็อกซี่ ซึ่งเมื่อพร็อกซี่ได้รับแล้วก็จะก่อนจะทำการส่งต่อต่อไปก็ต้องทำการตรวจในฟิลด์ส่วนของ Via ก่อนซึ่งจะพบว่ามีแอดเดรส ของตนก็จะเอาออกก่อนที่จะส่งต่อต่อไป

```
SIP/2.0 180 Ringing
```

```
Via: SIP/2.0/UDP 100.101.102.103:5060;branch=z9hG4bKmp17a
To: Heisenberg <sip:werner.heisenberg@munich.de>;tag=314159
From: E. Schroedinger <sip:schroed5244@aol.com>;tag=42
Call-ID: 10@100.101.102.103
CSeq: 1 INVITE
Contact: <sip:werner.heisenberg@200.201.202.203>
Content-Length: 0
```

การใช้ฟิลด์ Via ในส่วนเฮดเดอร์ในการค้นหาและส่งต่อซิพเมสเสจนั้นช่วยลดความซับซ้อนในการส่งต่อรีเควสที่ต้องใช้การค้นหาในฐานข้อมูลของพร็อกซี่ในการต้นหาเส้นทาง แต่การตอบรับ (response) ไม่ต้องการเพราะว่าใช้ Via ด้วยวิธีการนี้ทำให้แน่ใจได้ว่าการตอบรับนั้นๆจะกลับผ่านไปทางเดิมเสมอ ดังนั้นเมื่อยอมรับการรับสาย Heisenberg จึงส่ง 200 OK ดังนี้

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP proxy.munich.de:5060;branch=z9hG4bK83842.1
;received=100.101.102.105
Via: SIP/2.0/UDP 100.101.102.103:5060;branch=z9hG4bKmp17a
To: Heisenberg <sip:werner.heisenberg@munich.de>;tag=314159
From: E. Schroedinger <sip:schroed5244@aol.com>;tag=42
Call-ID: 10@100.101.102.103
CSeq: 1 INVITE
Contact: <sip:werner.heisenberg@200.201.202.203>
Content-Type: application/sdp
Content-Length: 159
v=0
o=heisenberg 2890844526 2890844526 IN IP4 200.201.202.203
s=Phone Call
c=IN IP4 200.201.202.203
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

และพร็อกซี่จะทำการส่งต่อต่อไปหลังจากได้นำฟิลด์ Via ของตนเองออกไปแล้วเช่นเดียวกัน

```
SIP/2.0 200 OK
```

```
Via: SIP/2.0/UDP 100.101.102.103:5060;branch=z9hG4bKmp17a
To: Heisenberg <sip:werner.heisenberg@munich.de>;tag=314159
From: E. Schroedinger <sip:schroed5244@aol.com>;tag=42
Call-ID: 10@100.101.102.103
CSeq: 1 INVITE
Contact: <sip:werner.heisenberg@200.201.202.203>
Content-Type: application/sdp
Introduction to SIP 29
Content-Length: 159
v=0
o=heisenberg 2890844526 2890844526 IN IP4 200.201.202.203
c=IN IP4 200.201.202.203
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

และจากการที่ในส่วนฟิลด์ Contact นั้น ด้วย SIP URI ของ Heisenberg ใน 200 OK นั้น ทำให้ Schroedinger สามารถส่ง ACK โดยตรงไปที่ Heisenberg โดยไม่ต้องผ่านทางพร็อกซี่ได้

```
ACK sip:werner.heisenberg@200.201.202.203 SIP/2.0
Via: SIP/2.0/UDP 100.101.102.103:5060;branch=z9hG4bKka42
Max-Forwards: 70
To: Heisenberg <sip:werner.heisenberg@munich.de>;tag=314159
From: E. Schroedinger <sip:schroed5244@aol.com>;tag=42
Call-ID: 10@100.101.102.103
CSeq: 1 ACK
Content-Length: 0
```

แสดงให้เห็นว่าจริงๆแล้วพร็อกซี่ซุร์ฟเวอร์นั้นไม่ได้อยู่ในการเรียกสายจริงๆ เพียงแต่ช่วยอำนวยความสะดวกให้กับปลายทางทั้งสองในการค้นหาที่อยู่และติดต่อซึ่งกันและกัน

เส้นทางการส่งสัญญาณของซิพนั้นเป็นอิสระทั้งหมดจากเส้นทางของการส่งมีเดียในทางโทรศัพท์นั้นจะเรียกว่าแยกกันระหว่างช่องทางควบคุมและช่องทางสื่อสาร

เมื่อมีเดียเซสชั่นจบลงเมื่อ Heisenberg ส่ง BYE request

```
BYE sip:schroed5244@pc33.aol.com SIP/2.0
Via: SIP/2.0/UDP 200.201.202.203:5060;branch=z9hG4bK4332
```

```
Max-Forwards: 70
To: E. Schroedinger <sip:schroed5244@aol.com>;tag=42
From: Heisenberg <sip:werner.heisenberg@munich.de>;tag=314159
Call-ID: 10@100.101.102.103
CSeq: 2000 BYE
Content-Length: 0
```

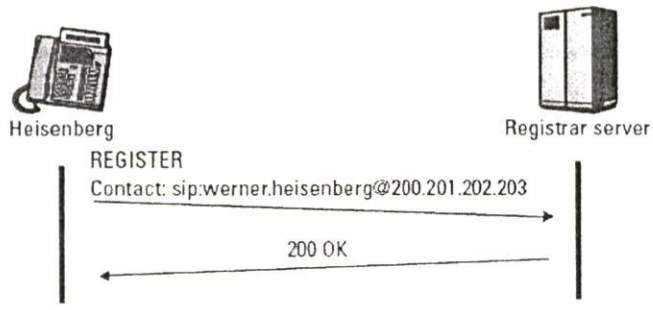
สังเกตว่าฟิลด์ CSEQ ของ Heisenbergนั้นคือ 2000 เนื่องจากแล้วแต่การตั้งค่าของตัวอุปกรณ์ซิพเอง

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP 200.201.202.203:5060;branch=z9hG4bK4332
To: E. Schroedinger <sip:schroed5244@aol.com>;tag=42
From: Heisenberg <sip:werner.heisenberg@munich.de>;tag=314159
Call-ID: 10@100.101.102.103
CSeq: 2000 BYE
Content-Length: 0
```

ในตัวอย่างต่อไปจะตอบคำถามที่ว่าฐานข้อมูลของพร็อกซี่นั้นมีไอพีแอดเดรสปัจจุบันของ Heisenberg ได้อย่างไร ซึ่งกลไกนั้นก็คือ REGISTRATION

2.1.3 ตัวอย่าง SIP Registration

ตัวอย่างนี้แสดงในรูปที่ 2.3 Heisenberg ส่ง SIP REGISTER request ไปที่ซิพเซิร์ฟเวอร์ชนิดหนึ่งที่รู้จักกันว่า Registrar Server ซึ่ง SIP registrar server รับเมสเสจเพื่อใช้เป็นข้อมูลเพื่ออัพเดทฐานข้อมูลที่จะถูกใช้โดยพร็อกซี่เพื่อค้นหาเส้นทางให้กับซิพรีเควสส่วนประกอบในเฮดเดอร์ของ REGISTER คือ SIP URI ของ Heisenberg ซึ่งเป็นแอดเดรสที่เป็นที่รู้จักกันดี อาจเป็นแอดเดรสที่พิมพ์ลงในนามบัตร และส่วนประกอบอีกส่วนคือ Contact URI ซึ่งใช้แทนอุปกรณ์ในขณะนั้น (และแทนไอพีแอดเดรส) ที่ผู้ใช้ Heisenberg ใช้อยู่ โดย Registrar จะผูก SIP URI ของ Heisenberg เข้ากับพีแอดเดรสของอุปกรณ์และเก็บไว้ในฐานข้อมูลที่สามารถถูกใช้ได้ ยกตัวอย่างเช่น proxy server ในรูปที่ 2.2 ที่รู้ถึงที่อยู่ของ Heisenberg



รูปที่ 2.3 ตัวอย่าง SIP registration

การ Registration นั้นในความเป็นจริงแล้วไม่มีในเครือข่ายโทรศัพท์ แต่มันคล้ายคลึงกันกับการ Registration ของโทรศัพท์เคลื่อนที่ที่เมื่อมันเปิด โทรศัพท์เคลื่อนที่จะส่ง Identity ไปที่ base station (BS) ซึ่งจะส่งต่อที่อยู่และหมายเลขโทรศัพท์ของโทรศัพท์นั้นๆไปที่ home location register (HLR) เมื่อ mobile switching center (MSC) ได้รับการเรียกสายเข้า มันจะไปถามที่ HLR เพื่อเอาที่อยู่ปัจจุบันของโทรศัพท์เคลื่อนที่เครื่องนั้น

REGISTER message ที่ถูกส่งโดย Heisenberg ไปที่ SIP registrar server มีรูปแบบดังนี้

```
REGISTER sip:registrar.munich.de SIP/2.0
Via: SIP/2.0/UDP 200.201.202.203:5060;branch=z9hG4bKus19
Max-Forwards: 70
To: Werner Heisenberg <sip:werner.heisenberg@munich.de>
From: Werner Heisenberg <sip:werner.heisenberg@munich.de>
;tag=3431
Call-ID: 23@200.201.202.203
CSeq: 1 REGISTER
Contact: sip:werner.heisenberg@200.201.202.203
Content-Length: 0
```

Registrar server จะตอบรับเมื่อทำการ Registration สำเร็จเรียบร้อยโดยการส่ง 200 OK กลับไปที่ Heisenberg

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP 200.201.202.203:5060;branch=z9hG4bKus19
To: Werner Heisenberg <sip:werner.heisenberg@munich.de>;tag=8771
From: Werner Heisenberg <sip:werner.heisenberg@munich.de>
;tag=3431
Call-ID: 23@200.201.202.203
```

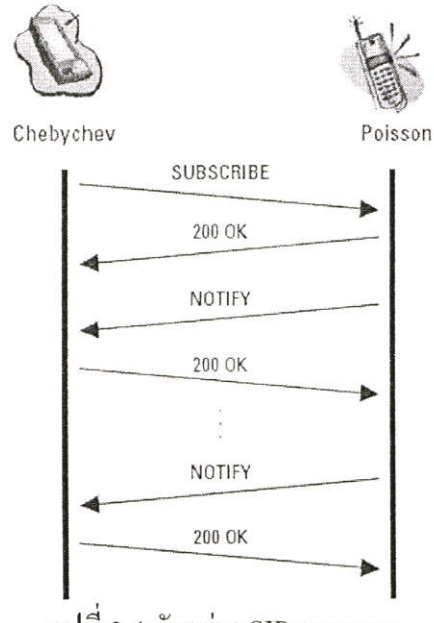
```
CSeq: 1 REGISTER
Contact: <sip:werner.heisenberg@munich.de>;expires=3600
Content-Length: 0
```

Contact URI ที่ส่งกลับมาจะมีฟิลด์ Expires จะชี้ว่าการ Registration นั้นมีอายุอยู่นานเท่าไหร่ ในกรณีนี้คือ 1 ชั่วโมง (3600 วินาที) ถ้า Heisenberg ต้องการ Registration ยังคงอยู่ในช่วงระยะเวลาหนึ่งก็ต้องส่ง REGISTER request อื่นไปอีกภายในช่วงก่อนหมดอายุ

โดยทั่วไปแล้วการ Registration จะกระทำโดยอัตโนมัติเมื่ออุปกรณ์ซิพนั้นๆเปิดเครื่องหรือภายในทุกๆช่วงเวลาก่อนที่จะหมดอายุซึ่งจะกำหนดโดย Registrar server ในการ Registration นั้นหลายๆอุปกรณ์อาจจะ Register ภายใต้ SIP URI เดียวกันได้ โดยพร็อกซี่นั้นอาจจะส่งต่อไปที่อุปกรณ์ทั้งคือหรือตัวใดตัวหนึ่งตามลำดับก็ได้ และการ Register ที่เพิ่มเติมเข้ามาจะเป็นการลบล้างการ Register ก่อนหน้านี้ด้วย

2.1.4 SIP Presence

ตัวอย่างนี้แสดงการใช้งานซิพในแอพพลิเคชั่นที่เป็น Presence คือข้อมูล Presence สามารถส่งสถานะของผู้ใช้หรือของอุปกรณ์ในขณะนั้นได้ มันสามารถถูกใช้ได้ง่ายๆเพื่อบอกสถานะของผู้ใช้ว่ากำลัง sign in (log on) อยู่หรือไม่ ว่ากำลัง idle หรือ away อยู่ สำหรับอุปกรณ์เคลื่อนที่แล้วข้อมูล Presence สามารถประกอบไปด้วยที่อยู่ปัจจุบันในลักษณะของคู่อันดับหรือในลักษณะทั่วๆไปเช่น ในสำนักงาน, ระหว่างเดินทาง, หรือในแลป โปรโตคอล Presence นั้นโดยหลักแล้วจะเกี่ยวพันกับความสัมพันธ์ของอุปกรณ์กับข้อมูลที่กำลังแลกเปลี่ยนกันอยู่ สำหรับซิพนั้น SUBSCRIBE ถูกใช้เพื่อร้องขอสถานะหรืออัพเดท Presence จาก Presence server (Presentity) และ NOTIFY ถูกใช้เพื่อส่งข้อมูลนั้นๆที่ได้ร้องขอมา



รูปที่ 2.4 ตัวอย่าง SIP presence

```
SUBSCRIBE sip:poisson@probability.org SIP/2.0
Via SIP/2.0/TCP lecturehall21.academy.ru:5060
;branch=z9hG4bK348471123
Max-Forwards: 70
To: M. Poisson <sip:poisson@probability.org>
From: P. L. Chebychev <sip:chebychev@academy.ru>;tag=21171
Call-ID: 58dkfj34924lk34452k592520
CSeq: 3412 SUBSCRIBE
Allow-Events: presence
Allow: ACK, INVITE, CANCEL, BYE, NOTIFY, SUBSCRIBE, MESSAGE
Contact: <sip:pafnuty@lecturehall21.academy.ru;transport=tcp>
Event: presence
Content-Length: 0
```

Possion ยอมรับ SUBSCRIBE request โดยการส่ง 202 Accepted กลับไปยัง Chebychev

```
SIP/2.0 202 Accepted
Via SIP/2.0/TCP lecturehall21.academy.ru:5060
;branch=z9hG4bK348471123;received=19.34.3.1
To: M. Poisson <sip:poisson@probability.org>;tag=25140
From: P. L. Chebychev <sip:chebychev@academy.ru>;tag=21171
Call-ID: 58dkfj34924lk34452k592520
CSeq: 3412 SUBSCRIBE
Allow-Events: presence
Allow: ACK, INVITE, CANCEL, BYE, NOTIFY, SUBSCRIBE, MESSAGE
Contact: <sip:s.possion@dist.probability.org;transport=tcp>
Event: presence
Expires: 3600
Content-Length: 0
```

การ Subscription ที่แท้จริงนั้นจะเริ่มเมื่อ Poisson ส่ง NOTIFY แรกกลับไปที่ Chebychev

```
NOTIFY sip:pafnuty@lecturehall21.academy.ru SIP/2.0
Via SIP/2.0/TCP dist.probablilty.org:5060
;branch=z9hG4bK4321
```

```
Max-Forwards: 70
To: P. L. Chebychev <sip:chebychev@academy.ru>;tag=21171
From: M. Poisson <sip:poisson@probability.org>;tag=25140
Call-ID: 58dkfj34924lk34452k592520
CSeq: 1026 NOTIFY
Allow: ACK, INVITE, CANCEL, BYE, NOTIFY, SUBSCRIBE, MESSAGE
Allow-Events: dialog
Contact: <sip:s.possion@dist.probability.org;transport=tcp>
Subscription-State: active;expires=3600
Event: presence
```

อย่าลืมว่า NOTIFY นั้นถูกส่งภายในไดอะล็อกเดียวกันกับ SUBSCRIBE มันจึงใช้ฟิลด์ Call_ID และ Tag เดียวกัน

Chebychev ส่ง 200 OK กลับเพื่อเป็นการยืนยันว่าได้รับ NOTIFY เรียบร้อยแล้ว

```
SIP/2.0 200 OK
Via SIP/2.0/TCP dist.probablilty.org:5060
;branch=z9hG4bK4321;received=24.32.1.3
To: P. L. Chebychev <sip:chebychev@academy.ru>;tag=21171
From: M. Poisson <sip:poisson@probability.org>;tag=25140
Call-ID: 58dkfj34924lk34452k592520
CSeq: 1026 NOTIFY
Content-Length: 0
```

2.2 SIP Client and Server

2.2.1 SIP User Agents

อุปกรณ์ปลายทางที่สามารถใช้ซิพได้จะถูกเรียกว่า SIP User Agent แนวคิดหนึ่งของซิพคือเพื่อให้สามารถเปิดหรือสร้างเซสชั่นระหว่าง User Agents ได้ และ User Agent จะต้องรับอินพุทจากผู้ใช้และแสดงเป็นตัวแทนเพื่อสร้างและทำลายมีเดียเซสชั่นด้วยกันกับ User Agent อื่น ในกรณีทั่วไปแล้วผู้ใช้มักเป็นมนุษย์แต่ผู้ใช้อาจจะเป็นโปรโตคอลอื่นๆ User Agent ต้องสามารถสร้างมีเดียเซสชั่นกับ User Agent อื่นได้

2.2.2 Presence Agents

เป็นอุปกรณ์ซิพที่สามารถรองรับการทำ Subscription และสร้างสถานะ Notification ให้มีลักษณะเป็นซิพอีเวนท์ได้

Presence Agent สามารถเก็บข้อมูล Presence จากหลายๆอุปกรณ์ได้ ข้อมูล Presence สามารถมาจากการ REGISTER ของอุปกรณ์ซิพ, จากการประกาศข้อมูล Presence โดยอุปกรณ์ซิพหรือจากที่มาอื่นๆที่ไม่ใช่ซิพ

Presence Server คือเซิร์ฟเวอร์ที่บางครั้งจะแสดงเป็น Presence Agent และเก็บข้อมูล Presence และหลายๆครั้งแสดงเป็นพร็อกซี่, ส่งต่อ SUBSCRIBE ไปยัง Presence Agent อื่นๆ

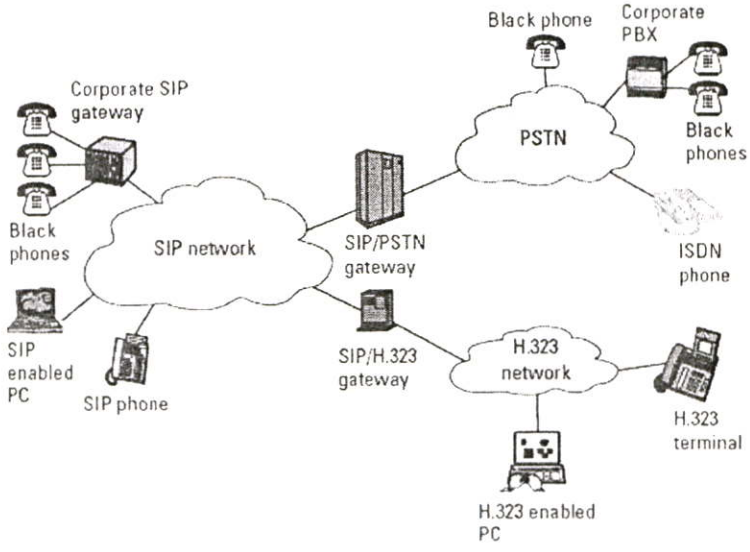
2.2.3 Back-to-Back User Agents

B2BUA เป็นอุปกรณ์ซิพประเภทหนึ่งที่รับซิพรีเควสและจะจัดรูปแบบใหม่ให้เป็นรีเควสใหม่ และเรสบอนส์ก็เช่นเดียวกันก็จะถูกจัดรูปแบบใหม่แล้วส่งกลับไปยังด้านตรงข้าม ในกรณีที่ 2 SIP UAs สามารถสื่อสารกันได้โดยไม่ต้องทราบถึง URI, ไอพีแอดเดรสและข้อมูลอื่นๆของกันและกันได้

2.2.4 SIP Gateways

ซิพเกทเวย์เป็นแอพพลิเคชั่นที่ซึ่งจะเชื่อมต่อระหว่างเครือข่ายซิพกับเครือข่ายอื่นๆ ในกรณีของซิพแล้วเกทเวย์เป็นเพียงแค่ชนิดพิเศษของ User Agent เท่านั้น ที่ซึ่ง User Agent กระทำเป็นโปรโตคอลอื่นๆไม่ใช่มนุษย์เกทเวย์จะตัดส่วนของสัญญาณและสามารถตัดส่วนมีเดียได้ ถึงแม้ว่านี่จะไม่ใช่กรณีที่เกิดขึ้นเสมอ ยกตัวอย่างเช่น SIP to H.323 gateway จะตัดสัญญาณซิพและแปลงสัญญาณเป็น H.323 แต่ SIP user agent และ H.323 terminal สามารถแลกเปลี่ยนข้อมูล RTP media โดยตรงซึ่งกันและกันได้โดยไม่ต้องผ่านเกทเวย์

รูปที่ 2.5 แสดงเครือข่ายซิพเชื่อมต่อผ่านเกทเวย์ด้วย PSTN และ H.323 เครือข่าย ซึ่งกรณีนี้จะสามารถทำงานได้ภายใต้มาตรฐาน SIP/H.323 internetworking

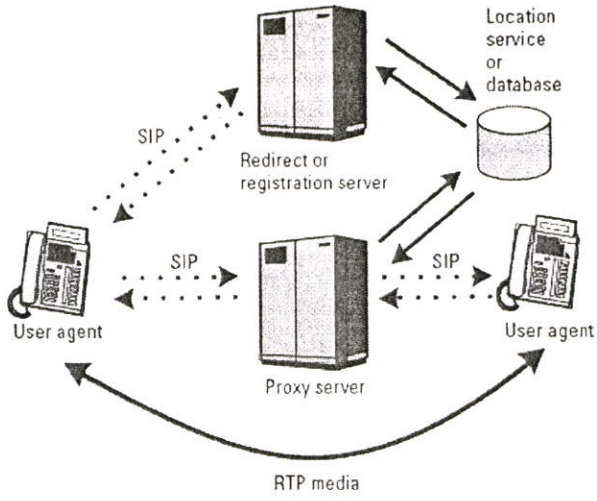


รูปที่ 2.5 เครือข่ายซิพด้วยเกทเวย์

ความแตกต่างระหว่าง User Agent และเกทเวย์คือจำนวนผู้ใช้ที่รองรับได้ ในขณะที่ User Agent โดยทั่วๆไปแล้วจะรองรับเพียงผู้ใช้เดียว แต่เกทเวย์สามารถรองรับได้เป็นร้อยหรือเป็นพันผู้ใช้ และเกทเวย์จะไม่ REGISTER ทุกๆผู้ใช้ที่มันรองรับแต่ User Agent จะเป็นอย่างนั้น

2.2.5 SIP Servers

ซิพเซิรฟเวอร์เป็นแอพพลิเคชั่นที่ยอมรับ SIP requests/respond กลับไป จะมีความแตกต่างระหว่างซิพเซิร์ฟเวอร์กับ User Agent server หรือไคลเอนต์เซิร์ฟเวอร์ของโปรโตคอลปกติ ที่ซึ่งอธิบายการทำงานในลักษณะของไคลเอนต์(ผู้เริ่มทำการีเควส) และเซิรฟเวอร์(ผู้เริ่มตอบสนองกรีเควส) ความจริงแล้วซิเพเซิร์ฟเวอร์ได้ถูกอิมพลีเมนต์ไว้หลายชนิด หรืออาจจะทำงานต่างลักษณะกันภายใต้เงื่อนไขต่างๆกัน เพราะเซิร์ฟเวอร์นั้นเตรียมบริการหรือการทำงานให้กับ User Agents ซึ่งพวกมันจะต้องรองรับทั้ง TCP, TLS และ UDP รูปที่ 2.6 แสดงการโต้ตอบกันของ User Agents, เซิร์ฟเวอร์และ Location Service



รูปที่ 2.6 การโต้ตอบกันของ SIP user agents, เซิร์ฟเวอร์และ Location Service

- Proxy Servers

SIP proxy server รับซิพรีเควสมาจาก User Agent หรือพร็อกซี่อื่นๆ และจะแสดงเป็นตัวแทนของ User Agent ในการส่งต่อหรือกระทำตามรีเควส ซึ่งพร็อกซี่นั้นไม่ใช่ B2BUA เพราะว่ามันถูกอนุญาติให้สามารถแก้ไขรีเควสและเรสบอนส์ได้ตามกฎที่ตั้งไว้ของ RFC3261 กฎนี้รักษาความเป็น transparency ของ end-to end ของสัญญาณซิพในขณะที่มันผ่านพร็อกซี่เซิร์ฟเวอร์

โดยทั่วไปแล้วพร็อกซี่เซิร์ฟเวอร์จะเข้าไปในฐานข้อมูลหรือ Location Service เพื่อประโยชน์ในการปฏิบัติตามรีเควส (ระบุจุดถัดไป) การเชื่อมต่อระหว่างพร็อกซี่และ location service นั้นไม่ได้ถูกระบุไว้ด้วยซิพพร็อกซี่ อาจจะใช้ฐานข้อมูลหลายชนิดเพื่อประโยชน์ในการปฏิบัติกับรีเควส

พร็อกซี่นั้นไม่ต้องเข้าถึงซิพรีเควสนอกจากการส่งต่อมันเท่านั้น ถ้าไม่รู้จักถึงชนิดของรีเควสที่เข้ามาก็จะสมมุติให้มันใช้ non-INVITE transaction model พร็อกซี่จะไม่เปลี่ยนลำดับของฟิลด์ในเฮดเดอร์หรือจะไม่ลบฟิลด์ในเฮดเดอร์

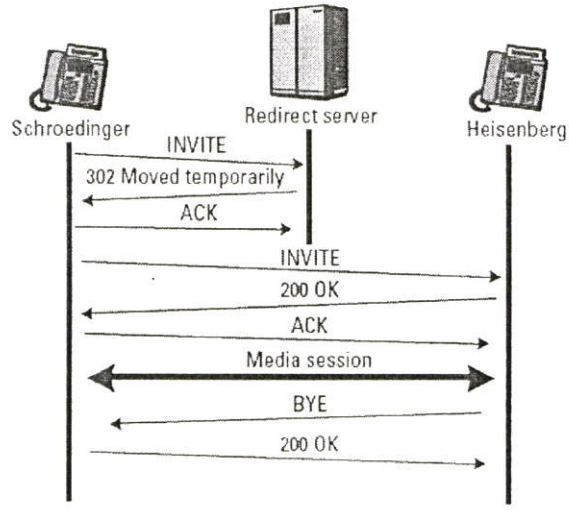
สิ่งที่แตกต่างกันระหว่างพร็อกซี่เซิร์ฟเวอร์และ User Agent หรือเกทเวย์มีอยู่ด้วยกันสามอย่างคือ 1) มันจะไม่สร้างรีเควสมันเพียงแต่จะเรสบอนส์ให้กับ User Agent 2) มันไม่มีความสามารถเกี่ยวกับมีเดีย และ 3) มันจะไม่ยุ่งกับส่วนของบอดี้

พร็อกซี่เซิร์ฟเวอร์สามารถเป็นได้ทั้งแบบ Stateless หรือ Stateful สำหรับ Stateless proxy server นั้นจะโพรเซสแต่ละซิพรีเควสหรือเรสบอนส์บนพื้นฐานของเนื้อหาภายในเมสเสจแต่ละเมสเสจจะถูกกระทำการต่างๆโดยที่จะไม่เก็บข้อมูลเกี่ยวกับมันไว้เลย Stateless proxy จะไม่มีการทำรีทรานมิสชั่นข้อความ แต่สำหรับ Stateful proxy server นั้นจะติดตามรีเควสและเรสบอนส์ที่เคยได้รับมาและใช้ข้อมูลของมันในการโพรเซสครั้งต่อๆไป อย่างเช่น มันจะมีการ รีท

รานมิสชั่นให้ในกรณีที่รีเควสที่มันได้ทำการส่งต่อไปไม่มีการตอบกลับมาภายในเวลาที่กำหนด ซึ่งปกติแล้วซิพพร้อกซี่จะเป็นประเภท Stateful

- Redirect Servers

Redirect server จะไม่ส่งต่อรีเควสแต่จะใช้ฐานข้อมูลหรือ Location Service ในการค้นหาผู้ใช้ ข้อมูลที่อยู่ของผู้ใช้จะถูกส่งกลับไปที่ผู้เรียกด้วยเรสบอนส์ที่เป็นคลาส redirection (3xx) โดยที่จะมี ACK หลังจบกระบวนการ รูปที่ 2.7 แสดงการไหลของเมสเสจที่เซิร์ฟเวอร์ใช้ redirection แทนที่พร็อกซี่เพื่อช่วยให้ Schroedinger หาที่อยู่ของ Heisenberg



รูปที่ 2.7 ตัวอย่าง redirect server

Redirect server ส่ง 302 Moved Temporarily เพื่อบอกให้ Schroedinger รู้ถึงที่อยู่ของ Heisenberg ผ่านทางฟิลด์ Contact

```
SIP/2.0 302 Moved Temporarily
Via: SIP/2.0/UDP 100.101.102.103:5060;branch=z9hG4bK54532
To: Heisenberg <sip:werner.heisenberg@munich.de>;tag=052500
From: E. Schroedinger <sip:schroed5244@aol.com>;tag=4313413
Call-ID: 9@100.101.102.103
CSeq: 1 INVITE
Contact: sip:werner.heisenberg@200.201.202.203
Content-Length: 0
```

Schroedinger ทำการตอบกลับด้วย ACK

```
ACK sip:werner.heisenberg@munich.de SIP/2.0
Via: SIP/2.0/UDP 100.101.102.103:5060;branch=z9hG4bK54532
```

```
Max-Forwards: 70
To: Heisenberg <sip:werner.heisenberg@munich.de>;tag=052500
From: E. Schroedinger <sip:schroed5244@aol.com>;tag=4313413
Call-ID: 9@100.101.102.103
CSeq: 1 ACK
Content-Length: 0
```

- Registration Servers

SIP registration server รู้จักกันในชื่อ Registrar ซึ่งจะรับเฉพาะ SIP REGSTER request เท่านั้น สำหรับรีเควสอื่นๆจะส่ง 501 Not Implemented เป็นเรสบอนส์กลับไป ในการรีเควสเพื่อ Registration นั้นฟิลด์ To ในเฮดเดอร์จะเป็นชื่อที่จะถูก register และฟิลด์ Contact จะเป็นชื่อหรือแอดเดรสอีกอันหนึ่ง คือ registration server จะสร้างการเชื่อมต่อชั่วคราวระหว่าง AOI URI กับฟิลด์ TO และ device URI กับฟิลด์ CONTACT

2.3 SIP Request Messages

ซิพรีเควสหรือเมธอดเป็น "verbs" ในโปรโตคอล คือพวกมันจะร้องขอให้เกิดการกระบวนการต่างๆเพื่อให้กับ User Agent หรือเซิร์ฟเวอร์ มี INVITE, REGISTER, BYE, ACK, CANCEL และ OPTIONS เป็นเมธอดเริ่มแรกใน SIP RFC3261 และ REFER, SUBSCRIBE, NOTIFY, MESSAGE, UPDATE, INFO และ PRACK ถูกอธิบายแยกไว้ในอีก RFCs อื่น

2.3.1 INVITE method

INVITE ถูกใช้เพื่อสร้างมีเดียเซสชั่นระหว่าง User Agent การตอบรับหรือตอบกลับของ NIVITE มักจะเป็น ACK เสมอ ใน INVITE นั้นมักจะประกอบไปด้วยข้อมูลของมีเดียของผู้เรียก ซึ่งถ้าไม่มีข้อมูลส่วนนี้ใน INVITE ก็จะมีในส่วนของ ACK ของ UAC แต่ถ้าข้อมูลของมีเดียที่อยู่ใน ACK นั้นไม่สามารถถูกยอมรับได้แล้ว call party ต้องส่ง BYE เพื่อยกเลิกเซสชั่น (ไม่ใช้ CANCEL เนื่องจากเซสชั่นนั้นๆได้ถูกสร้างขึ้นมาแล้ว) มีเดียเซสชั่นนั้นจะถูกสร้างขึ้นเมื่อ INVITE, 200 OK และ ACK ได้ถูกแลกเปลี่ยนระหว่าง UAC และ UAS แล้ว INVITE จะสำเร็จสมบูรณ์จนกระทั่งเมื่อ BYE ถูกส่งเพื่อจบเซสชั่น

บางครั้ง re-INVITE จะถูกใช้เพื่อเปลี่ยนแปลงลักษณะของเซสชั่น ถ้า re-INVITE ถูกปฏิเสธหรือล้มเหลวแล้วเซสชั่นก็ยังคงอยู่ re-INVITE จะไม่ถูกส่งโดย UAC จนกระทั่งตอบรับสุดท้ายของ INVITE แรกนั้นถูกรับแล้ว ตรงข้ามกันกับ UPDATE ที่สามารถถูกส่งได้

2.3.2 REGISTER method

REGISTER นั้นถูกใช้โดย User Agent เพื่อให้เครือข่ายซิพของมันรู้ถึง Contact URI (ไอพีแอดเดรส) ปัจจุบันของมันและ URI นี้จะมีรีเควสที่ถูกค้นหาเส้นทางมาถึง Contact นี้ การ Registrationนั้นไม่จำเป็นสำหรับ User Agent เพื่อให้สามารถใช้พร็อกซี่เซิร์ฟเวอร์เพื่อส่งการเรียกสายออกไป แต่มันจำเป็นสำหรับ User Agent ที่ต้อง Register เพื่อให้สามารถรับการเรียกสายที่เข้ามาจากพร็อกซี่ได้

2.3.3 BYE method

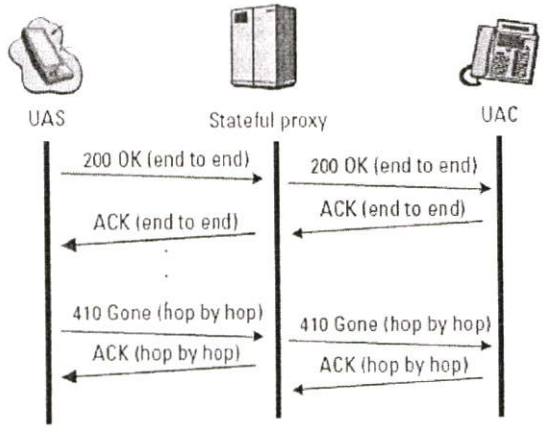
ถูกใช้เพื่อตัดมีเดียเซสชั่นที่ได้สร้างไว้ BYE จะถูกส่งจาก User Agent ที่ร่วมในเซสชั่นนั้นๆเท่านั้น จะไม่ถูกส่งโดยพร็อกซี่หรือเครื่องมือร่วมอื่นๆ เป็นเมธอดที่เป็น end-to-end ที่เรสบอนส์ของมันจะถูกส่งมาจากอีก User Agent หนึ่งเท่านั้น User Agent จะตอบรับกลับด้วย 481 Dialog/Transaction Does Not Exist เมื่อ BYE ในกรณีที่ไม่รู้ไดอะล็อกนั้นๆ

2.3.4 ACK method

ACK ถูกใช้เพื่อเป็น acknowledge สุดท้ายเพื่อตอบกลับ INVITE การตอบกลับสุดท้ายของรีเควสอื่นๆจะไม่ใช่ acknowledge แต่จะเป็น 2xx, 3xx, 4xx หรือ 5xx

ACK อาจจะบรรจุ "application/sdp" ไว้ในบอดี้ในกรณีที่ INVITE เริ่มแรกนั้นไม่มี SDP และ ACK จะไม่ถูกใช้เพื่อแก้ไขรายละเอียดของมีเดียที่ได้ถูกส่งไปแล้วใน INVITE เริ่มแรก (ในกรณีนี้จะใช้ re-INVITE)

ACK เป็น end-to-end แต่สำหรับเรสบอนส์สุดท้ายอื่นๆอย่าง 2xx แล้วจะเป็นแบบ hop-by-hop เมื่อใช้ Stateful Proxies ความแตกต่างระหว่าง ACK ที่ส่งแบบ hop-by-hop กับแบบ end-to-end นั้นจะถูกแสดงในรูปที่ 2.8



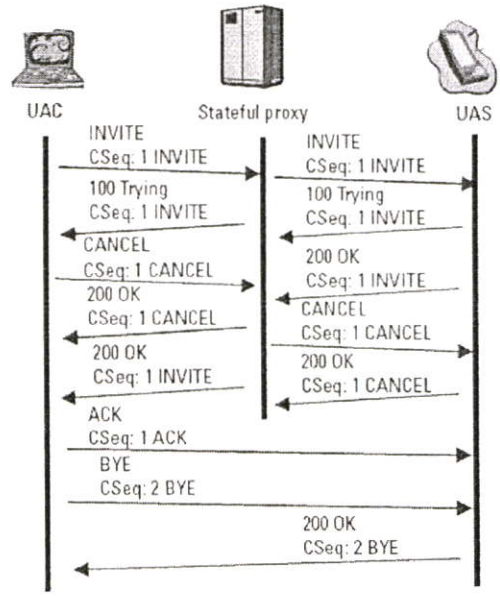
รูปที่ 2.8 ACK แบบ end-to-end และ hop-by-hop

2.3.5 CANCEL method

ถูกใช้เพื่อตัดการเรียกสายหรือการค้นหาเส้นทางเรียกสายที่ยังค้างอยู่ (การเรียกสายที่ยังไม่เสร็จ ยังไม่ได้สร้างเป็นเซสชั่น) มันสามารถถูกสร้างได้จากทั้ง User Agents และ Proxy server ที่ได้เตรียมเรสบอนส์ 1xx ไว้แต่ว่ายังไม่ได้รับเรสบอนส์สุดท้าย (2xx)

เมื่อพร็อกซี่ได้รับ CANCEL ก็จะส่งต่อมันไปยังตามชุดเส้นทางเดียวกันกับที่มันได้ส่ง INVITE เริ่มแรกไป พร็อกซี่จะมักจะไม่เรสบอนส์เพื่อส่งต่อ CANCEL สำหรับ User Agent นั้นจะต้องยืนยันการยกเลิกด้วย 200 OK และตอบ INVITE กลับด้วย 487 Request Terminated

ถ้าได้รับเรสบอนส์สุดท้าย (2xx) เรียบร้อยแล้ว User Agent จะต้องส่ง BYE เพื่อตัดเซสชั่น นี่เป็นกรณีที่เกิดการสวนกันของ CANCEL และเรสบอนส์สุดท้ายขึ้นภายในเครือข่าย จะแสดงดังรูปที่ 2.9 คือในตัวอย่างนี้ CANCEL และ 200 OK เดินทางสวนกันระหว่างพร็อกซี่และ UAS ทำให้พร็อกซี่ยังต้องตอบ CANCEL ด้วย 200 OK แต่ก็ต้องส่งต่อ 200 OK ตอบกลับให้กลับ INVTE สำหรับ 200 OK ที่เป็นการตอบกลับของ CANCEL ที่ถูกส่งโดยพร็อกซี่นั้นมีความหมายเพียงแค่ได้รับ CANCEL เรียบร้อยแล้ว และกำลังส่งต่อมันอยู่ UAC ต้องเตรียมตัวที่จะรับเรสบอนส์สุดท้ายอยู่ แต่จะไม่มี 487 response ในกรณีนี้เพราะเซสชั่นถูกยกเลิกโดยการที่ UAC ส่ง ACK และ BYE เพื่อรับ 200 OK



รูปที่ 2.9 กรณีที่เกิดการแย่งชิงกันในการยกเลิกการเรียกสาย

2.3.6 OPTIONS method

ถูกใช้เพื่อถาม User Agent หรือ server ถึงความสามารถของมันหรือเพื่อค้นหาสิ่งที่ยังสามารถใช้ได้อยู่ proxy จะไม่สร้าง OPTIONS ส่วน User Agent หรือเซิร์ฟเวอร์จะตอบกลับ OPTIONS เช่นเดียวกันกับ INVITE คือจะตอบ 2xx กลับมาเพื่อบอกถึงความสามารถที่มีด้วยฟิลด์ Allow, Accept, Accept-Encoding, Accept-Language และ Supported

พร็อกซี่จะตรวจสอบว่า OPTIONS นี้เป็นของมันหรือไม่โดยการดูจากฟิลด์ Request-URI ซึ่งถ้า Request-URI นั้นเป็นแอดเดรสของพร็อกซี่ก็หมายถึง OPTIONS นี้เป็นของมัน แต่ถ้าไม่ใช่มันก็จะส่งต่อ OPTIONS ไป

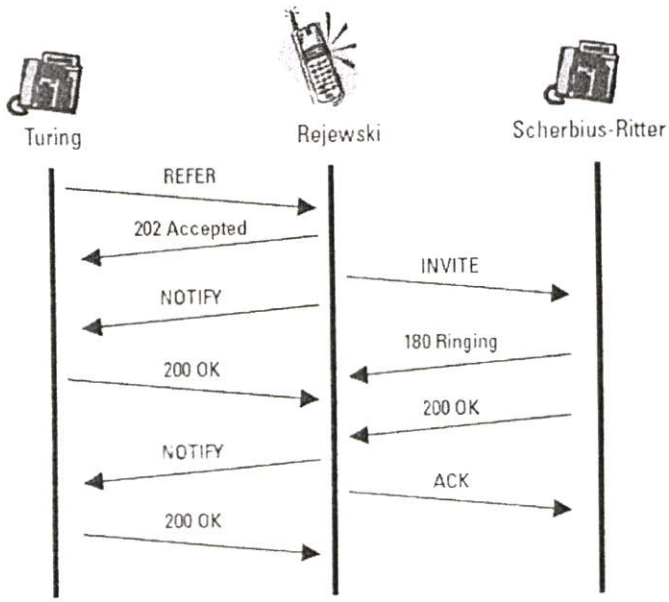
```
OPTIONS sip:user@carrier.com SIP/2.0
Via: SIP/2.0/UDP cavendish.kings.cambridge.edu.uk
;branch=z9hG4bK1834
Max-Forwards:70
To: <sip:user@proxy.carrier.com>
From: J.C. Maxwell <sip:james.maxwell@kings.cambridge.edu.uk>
;tag=34
Call-ID: 9352812@cavendish.kings.cambridge.edu.uk
CSeq: 1 OPTIONS
Content-Length: 0
```

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP cavendish.kings.cambridge.edu.uk;tag=512A6
;branch=z9hG4bK0834 ;received=192.0.0.2
To: <sip:user@proxy.carrier.com>;tag=432
From: J.C. Maxwell <sip:james.maxwell@kings.cambridge.edu.uk>
;tag=34
Call-ID: 9352812@cavendish.kings.cambridge.edu.uk
CSeq: 1 OPTIONS
Allow: INVITE, OPTIONS, ACK, BYE, CANCEL, REFER
Accept-Language: en, de, fr
Content-Length: ...
Content-Type: application/sdp
v=0
etc...
```

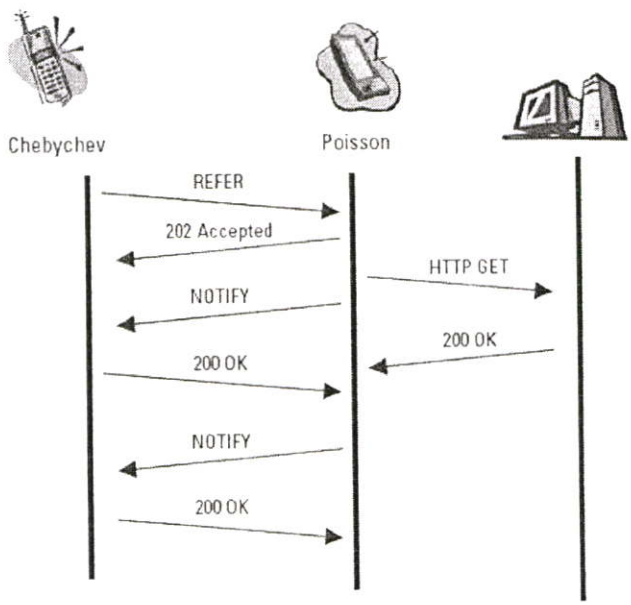
2.3.7 REFER method [10]

ถูกใช้โดย User Agent เพื่อขอให้ User Agent อื่นๆนั้นเข้า URI ซึ่งจะถูกระบุไว้ในฟิลด์ Refer-To ในเฮดเดอร์

REFER นั้นสามารถถูกส่งได้ทั้งภายในและภายนอกไดอะล็อก รูปแบบทั่วๆไปถูกแสดงในรูปที่ 2.10 ในตัวอย่างนี้ UAC ส่ง REFER ให้ UAS และจะตอบรับด้วย 202 Accepted ซึ่งเรสบอนส์ที่ใช้ในตอบรับนี้จะตอบกลับทันทีโดยที่ไม่ต้องรอให้ทำตามรีเควสนั้นจนเสร็จ



รูปที่ 2.10 ตัวอย่าง REFER โดยเป็น Refer-to URI

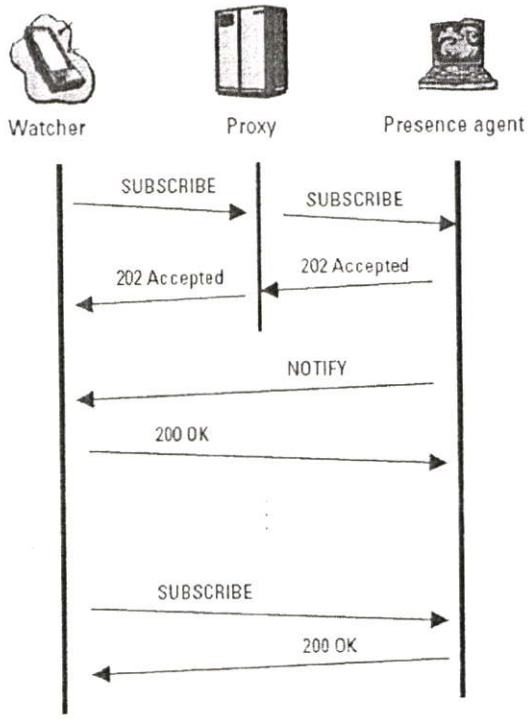


รูปที่ 2.11 ตัวอย่าง REFER เพื่อให้เปิดหน้าเว็บ

2.3.8 SUBSCRIBE method [11]

ถูกใช้โดย User Agent เพื่อให้ได้รับ NOTIFY ถึงเหตุการณ์เฉพาะ SUBSCRIBE สามารถถูกเคลย์ใหม่ได้โดยการส่ง SUBSCRIBE ใหม่อื่นภายในไดอะล็อกก่อนหมด Expires

สำหรับเซิร์ฟเวอร์จะยอมรับ SUBSCRIBE ได้โดยการส่ง 200 OK กลับมา หากมี 202 Accepted กลับมาจะหมายถึงเพียงแค่ว่าเซิร์ฟเวอร์รับทราบหรือเข้าใจแล้ว



รูปที่ 2.12 ตัวอย่าง SUBSCRIBE และ NOTIFY

2.3.9 NOTIFY method [11]

ถูกใช้โดย User Agent เพื่อนำส่งข้อมูลเกี่ยวกับเหตุการณ์เฉพาะ NOTIFY จะถูกส่งภายในไดอะล็อกเมื่อมี SUBSCRIBE

NOTIFY จะได้รับการตอบกลับเป็น 200 OK เพื่อบอกว่าได้รับแล้ว แต่ถ้าได้รับ 481 Dialog/Transaction Does Not Exist แล้ว SUBSCRIBE นั้นจะถูกตัดโดยอัตโนมัติและจะไม่มีการส่ง NOTIFY อีก

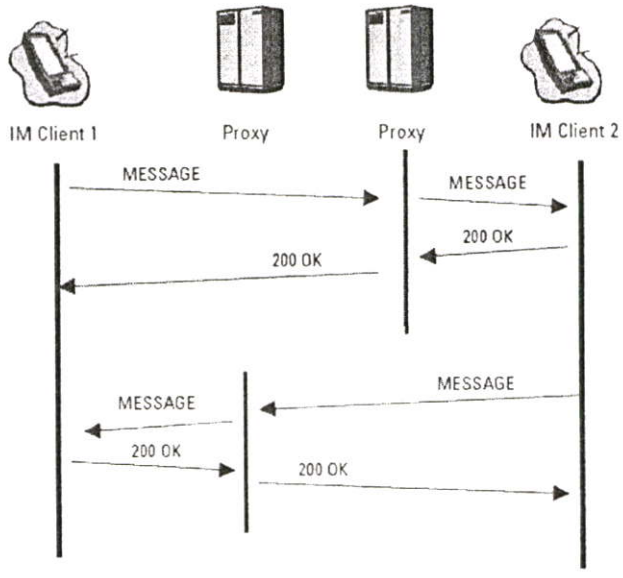
NOTIFY จะถูกส่งเมื่อเริ่มและเมื่อจบการ Subscription

2.3.10 MESSAGE method [12]

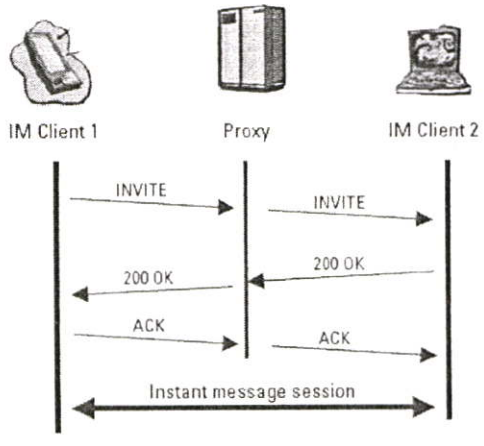
ถูกใช้เพื่อส่ง Instant Message (IM) ของซิพซึ่งมักจะเป็นข้อความสั้นๆที่ทำการแลกเปลี่ยนกันในลักษณะที่เกือบจะเป็นเรลไทม์ MESSAGEs อาจจะถูกส่งภายในหรือภายนอกไดอะล็อกได้ แต่พวกมันจะไม่สามารถสร้างไดอะล็อกขึ้นมาเองได้ ตัวข้อความจริงๆจะอยู่ในบอดี้เป็น MIME attachment

MESSAGE โดยทั่วไปแล้วจะได้รับ 200 OK เป็นการตอบรับเพื่อชี้ให้เห็นว่าข้อความได้ไปถึงจุดหมายปลายทางเรียบร้อยแล้ว IM ไม่ควรจะถูกส่งเป็นข้อความโดยเอาไว้ใน 200 OK แต่

ควรจะแยกเป็น MESSAGE ไว้ สำหรับการที่ได้รับ 202 Accepted กลับมาหมายถึงมีการ store-and forward ของอุปกรณ์ก่อนที่จะส่งไปยังปลายทาง



รูปที่ 2.13 ตัวอย่าง SIP instant message



รูปที่ 2.14 การใช้ซิพเพื่อสร้างเซสชั่นของ Instant Message

2.3.11 INFO method [13]

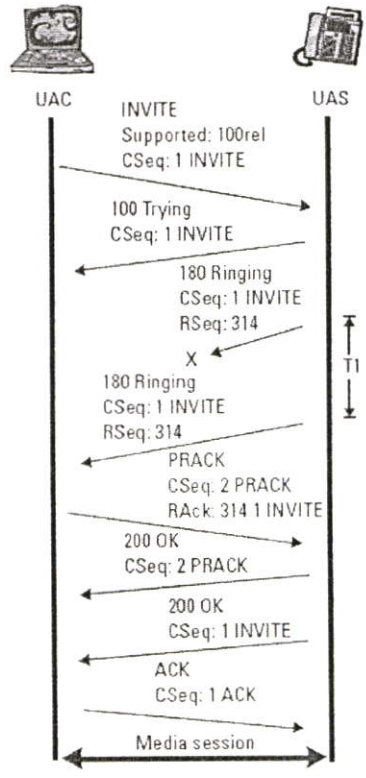
ถูกใช้โดย User Agent เพื่อส่งข้อมูลของการเรียกสายไปยังอีก User Agent ที่ซึ่งมีมีเดียเซสชั่นอยู่แล้ว แตกต่างจาก re-INVITE คือมันจะไม่เปลี่ยนลักษณะของมีเดียของการเรียกสายนั้นๆ INFO เป็น end-to-end จะไม่ถูกแก้ไขโดยพร็อกซี่

ปกติแล้ว INFO จะมีส่วนบอดี้ของเมสเสจซึ่งจะบรรจุข้อมูลของสัญญาณไว้หรือ midcall event

2.3.12 PRACK method [14]

ใช้เพื่อเป็น acknowledge ให้กับเรสปอนส์แบบ provisional (1xx) เนื่องจาก 2xx, 3xx, 4xx, 5xx และ 6xx ที่เป็นเรสปอนส์ให้กับ INVITEs นั้นใช้ ACK อย่างไรก็ตามในกรณีที่เรสปอนส์แบบ provisional อย่างเช่น 180 Ringing เกิดเหตุการณ์ผิดปกติขึ้น (อย่างเช่นส่งไปไม่ถึง) มันจึงอาจจะจำเป็นที่จะต้องมีการยืนยันให้กับเรสปอนส์แบบ provisional ซึ่ง PRACK จะตอบกลับๆทุกๆเรสปอนส์ provisional ยกเว้น 100 Trying

PRACK จะถูกสร้างโดย UAC เมื่อได้รับเรสปอนส์แบบ provisional ที่มีฟิลด์ RSeq และ Supported: 100rel ในเฮดเดอร์ PRACK จะสะท้อนหมายเลข RSeq และ CSeq กลับไปในเฮดเดอร์ การทำงานของ PRACK จะแสดงไว้ดังรูปที่ 2.15 คือ UAC ส่ง 180 Ringing โดยที่มีฟิลด์ RSeq ในเฮดเดอร์ เมื่อไม่ได้รับ PRACK จาก UAC ภายในเวลาที่กำหนดก็จะส่งเรสปอนส์กลับไปอีกครั้ง จนกระทั่งได้รับ PRACK แล้วจึงจะเลิกทำการส่งซ้ำ และ 200 OK ตอบกลับ PRACK จึงจะเป็นการหยุดส่ง PRACK ซ้ำ การเรียกสายจะเสร็จสมบูรณ์เมื่อ UAC ส่ง ACK เพื่อตอบกลับ 200 OK

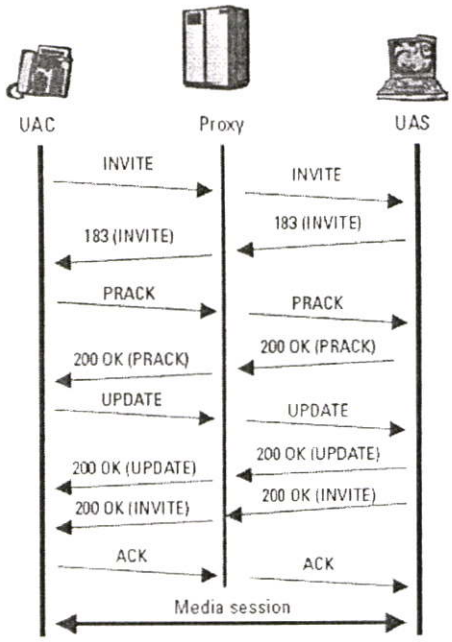


รูปที่ 2.15 การใช้ PRACK

การส่งซ้ำหรือรีทรานมิสชั่นนั้นจะใช้หลักการของ exponential backoff เพื่อรอเรสปอนส์สุดท้ายของ INVITE

2.3.13 UPDATE method [15]

ถูกใช้เพื่อแก้ไขสถานะของเซสชั่นโดยไม่ต้องเปลี่ยนสถานะของไดอะล็อก การใช้ UPDATE ก็อย่างเช่น mute หรือ hold



รูปที่ 2.16 ตัวอย่างการ UPDATE

2.4 SIP Responses Messages

ซิพเรสบอนส์เมสเสจถูกสร้างโดย UAS หรือซิพเซิร์ฟเวอร์เพื่อตอบกลับรีเควสที่ถูกสร้างโดย UAC มีซิพเรสบอนส์อยู่ 6 ประเภท 5 ประเภทแรกนั้นได้ยืมมาจาก HTTP และสุดท้ายถูกสร้างขึ้นจากซิพเอง

ถ้าไม่รู้จักรหัสของซิพเรสขอนส์นั้น UAC จะต้องแปลจากคลาสของเรสบอนส์นั้นๆ เช่น ไม่รู้จัก 599 Server Unplugged มันจะถูกแปลโดย User Agent เป็น 500 Server Failure

ตารางที่ 2.1 SIP Response Classes

Class	Description	Action
1xx	Informational	บ่งบอกสถานะของการเรียกสายก่อนเสร็จสมบูรณ์ จะเป็นข้อมูลแรกหรือเรสบอนส์แบบ provisional
2xx	Success	รีเควสนั้นได้เสร็จแล้ว มีไว้สำหรับเพื่อส่งให้ INVITE หรือACK เพื่อให้หยุดการส่งซ้ำ (รีทรานมิสชั่น) ของรีเควส
3xx	Redirection	เซิร์ฟเวอร์ส่งที่อยู่ที่เป็นไปได้กลับมา ไคลเอนต์ควรจะลองส่งรีเควสไปที่เซิร์เวอร์อื่นๆ
4xx	Client error	รีเควสไม่สำเร็จเนื่องจากเกิดข้อผิดพลาดขึ้นที่ไคลเอนต์ทำให้ไคลเอนต์อาจจะต้องลองทำรีเควสนั้นใหม่เพื่อให้ได้เรสบอนส์
5xx	Server failure	รีเควสไม่สำเร็จเนื่องจากเกิดข้อผิดพลาดขึ้นที่เซิร์ฟเวอร์ ทำให้รีเควสอาจจะต้องถูกลองทำใหม่ที่เซิร์ฟเวอร์อื่น
6xx	Global failure	รีเควสไม่สำเร็จ และรีเควสนั้นๆไม่ควจะถูกลองทำใหม่ที่เซิร์ฟเวอร์นี้หรือเซิร์ฟเวอร์ใดๆ

2.4.1 Information

ใช้เพื่อบอกถึงความก้าวหน้าในการเรียกสายและเป็น end-to-end ยกเว้น 100 Trying ที่ซึ่งอาจจะเป็น hop-by-hop สำหรับเรสบอนส์แบบ Information แรกที่ UAC ได้รับจะเป็นการบอกให้รีทรานมิสชั่นในการ INVITE และเรสบอนส์แบบ information นอกจากนี้จะไม่มีผลกับ INVITE เรสบอนส์แบบ Information นั้นเป็นแค่ออพชั่นที่ซึ่ง UAS อาจจะส่งเรสบอนส์สุดท้ายได้โดยที่ไม่ต้องส่งเรสบอนส์แบบ information เลย และในขณะที่ INVITE นั้นมี ACK เพื่อเป็นการยืนยัน แต่สำหรับเรสบอนส์แบบ information แล้วไม่มีการยืนยันใดๆยกเว้น PRACK และทุกๆตัวยกเว้น 100 Trying จะต้องมีฟิลด์ Contact URI และแสดงทุก Record-Route

- 100 Trying
- 180 Ringing
- 181 Call Is Being Forwarding
- 182 Call Queued
- 183 Session Progress

2.4.2 Success

เพื่อชี้ว่ารีเควสนั้นๆได้สำเร็จหรือเป็นที่ตกลงยอมรับแล้ว

- 200 OK
- 202 Accepted

2.4.3 Redirection

ถูกส่งโดย SIP redirect server เพื่อเป็นเรสปอนส์ให้กับ INVITE

- 300 Multiple Choice
- 301 Moved Permanently
- 302 Moved Temporarily
- 305 Use Proxy
- 380 Alternative Server

2.4.4 Client Error

ถูกใช้โดยเซิร์ฟเวอร์หรือ UAS เพื่อบอกว่ารีเควสนั้นๆไม่สามารถถูกทำให้สมบูรณ์ได้ มันจะระบุถึงสาเหตุที่ไคลเอนต์เกิดข้อผิดพลาดรวมทั้งวิธีแก้ไข ทำให้ UAC ไม่สามารถส่งรีเควสเดิมมาได้โดยไม่มีการปรับเปลี่ยนแก้ไข

- 400 Bad Request
- 401 Unauthorized
- 402 Payment Required
- 403 Forbidden
- 404 Not Found
- 405 Method Not Allowed
- 406 Not Acceptable
- 407 Proxy Authentication Required
- 408 Request Timeout
- 409 Conflict
- 410 Gone
- 411 Length Required
- 413 Request Entity Too Large
- 414 Request- URI Too Long
- 415 Unsupported Media Type
- 416 Unsupported URI Scheme
- 410 Bad Extension
- 421 Extension Required
- 422 Session Timer Interval Too Small
- 423 Interval Too Brief

- 428 Use Authentication Token
- 429 Use Authentication Token
- 480 Temporarily Unavailable
- 481 Dialog/Transaction Does Not Exist
- 482 Loop Detected
- 483 Too Many Hops
- 484 Address Incomplete
- 485 Ambiguous
- 486 Busy Here
- 487 Request Terminated
- 488 Not Acceptable Here
- 489 Bad Event
- 491 Request Pending
- 493 Request Undecipherable

2.4.5 Server Error

ใช้เพื่อบอกว่ารีเควสนั้นไม่สามารถถูกนำไปทำตามได้เพราะเกิดข้อผิดพลาดขึ้นที่เซิร์ฟเวอร์ แต่รีเควสสามารถถูกนำไปทำที่อื่นได้เพราะไม่มีข้อผิดพลาดเกิดขึ้นในรีเควส

- 500 Server Internal Error
- 501 Not Implemented
- 502 Bad Gateway
- 503 Service Unavailable
- 504 Gateway Timeout
- 505 Version Not Supported
- 513 Message Too Large

2.4.6 Global Error

ใช้เพื่อชี้ว่าเซิร์ฟเวอร์นั้นรู้ว่ารีเควสนั้นจะเกิดการล้มเหลวขึ้น ทำให้รีเควสไม่ควรจะถูกส่งไปที่อื่น

- 600 Busy Everywhere
- 603 Decline
- 604 Does Not Exist Anywhere
- 606 Not Acceptable

บทที่ 3

การออกแบบและพัฒนา

SIP Traffic Generator and Analyzer ได้พัฒนามาจากโอเพ่นซอร์สโดยได้ทำการเลือกจากที่มีเผยแพร่อยู่แล้วหลายๆตัว ซึ่งสุดท้ายได้ทำการเลือก SIPp [16] ขึ้นมาพัฒนาต่อ เพราะมีความยืดหยุ่นในด้านการมากกว่าตัวอื่นๆ ทั้งนี้ได้ทำการพัฒนาเพิ่มเติมโดยการพอร์ทจากที่ทำงานบนลีนุกซ์มาเป็นทำงานบนวินโดวส์ เพิ่มความสามารถให้ทำงานได้ใกล้เคียงกับมาตรฐานของซิพมากขึ้น ให้สามารถจำลองสถานการณ์เป็นผู้ใช้หลายๆคนได้ และได้เพิ่มสูตรการคำนวณเพื่อวัดค่าประสิทธิภาพทางเครือข่าย เช่น Erlang เข้ามา

3.1 เครื่องมือหรืองานที่เกี่ยวข้อง

SIP Traffic Generator and Analyzer ได้เลือกโอเพ่นซอร์สขึ้นมาเพื่อทำการพัฒนาต่อ ซึ่งกลุ่มของโอเพ่นซอร์สเริ่มแรกที่ได้นำมาศึกษาเพื่อเลือกนำไปพัฒนาต่อเหล่านั้นได้แก่ Sipsak [17], SipBomber [18] และ SIPp [16]

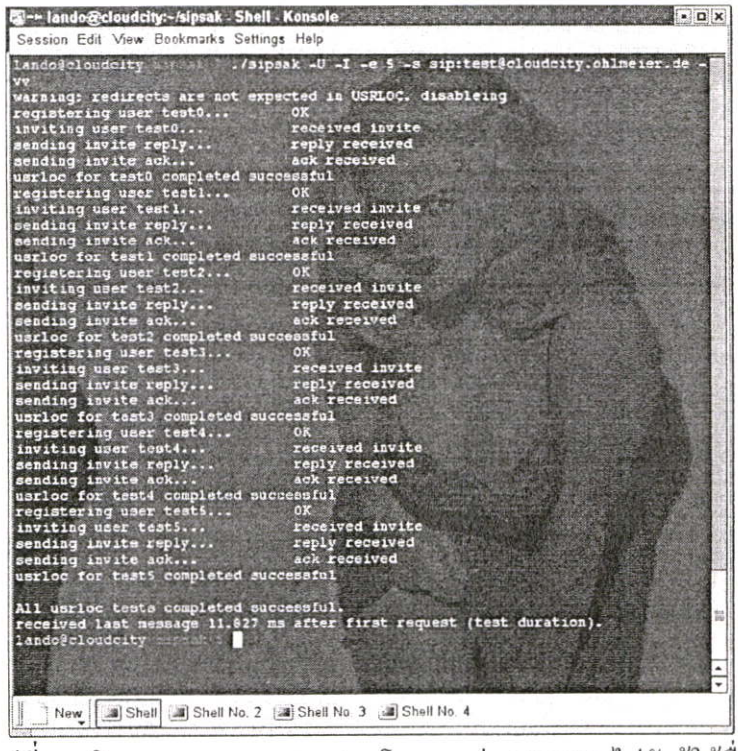
3.1.1 Sipsak 0.8.12

เป็นเครื่องมือที่เป็นคอมมานไลน์ขนาดเล็กที่มีไว้สำหรับผู้พัฒนาและผู้ดูแลของซิพแอพพลิเคชั่น มันสามารถถูกใช้เพื่อทดสอบอย่างง่ายๆบนซิพแอพพลิเคชั่น และอุปกรณ์ มีความสามารถดังนี้คือ

- ส่ง OPTIONS request
- ส่งไฟล์ข้อความโดยการบรรจุไว้ในซิพรีเควส
- ทำ Tracerout
- ทดสอบที่อยู่ของผู้ใช้
- ทดสอบการทำ flooding
- สุ่มลักษณะการทดสอบ
- ตอบโต้และปฏิบัติเป็นซิพเรสปอนส์
- ทำ authentication ด้วย MD5 และ SHA1
- เพิ่มเฮดเดอร์ฟิลด์เข้าไปในรีเควส
- ทำการเรียกสาย
- ทำงานได้ภายหลัง NAT
- สามารถส่งหรือให้ Contact แก่ Registrar ได้ (REGISTER)
- ส่งข้อความไปยังปลายทางใดๆที่เป็นซิพได้

- ใช้กระบวนการแบบมัลติเปิ้ลโพรเซสเพื่อสร้างโหลดให้แก่เซิร์ฟเวอร์ได้
- รองรับทั้ง UDP และ TCP

รูปที่ 3.1 ตัวอย่างการทดสอบอย่างง่ายของ OPTIONS request ของ Sipsak



รูปที่ 3.2 โหมดการทดสอบ usrloc โดยการส่ง INVITE ไปยังผู้ใช้อื่น

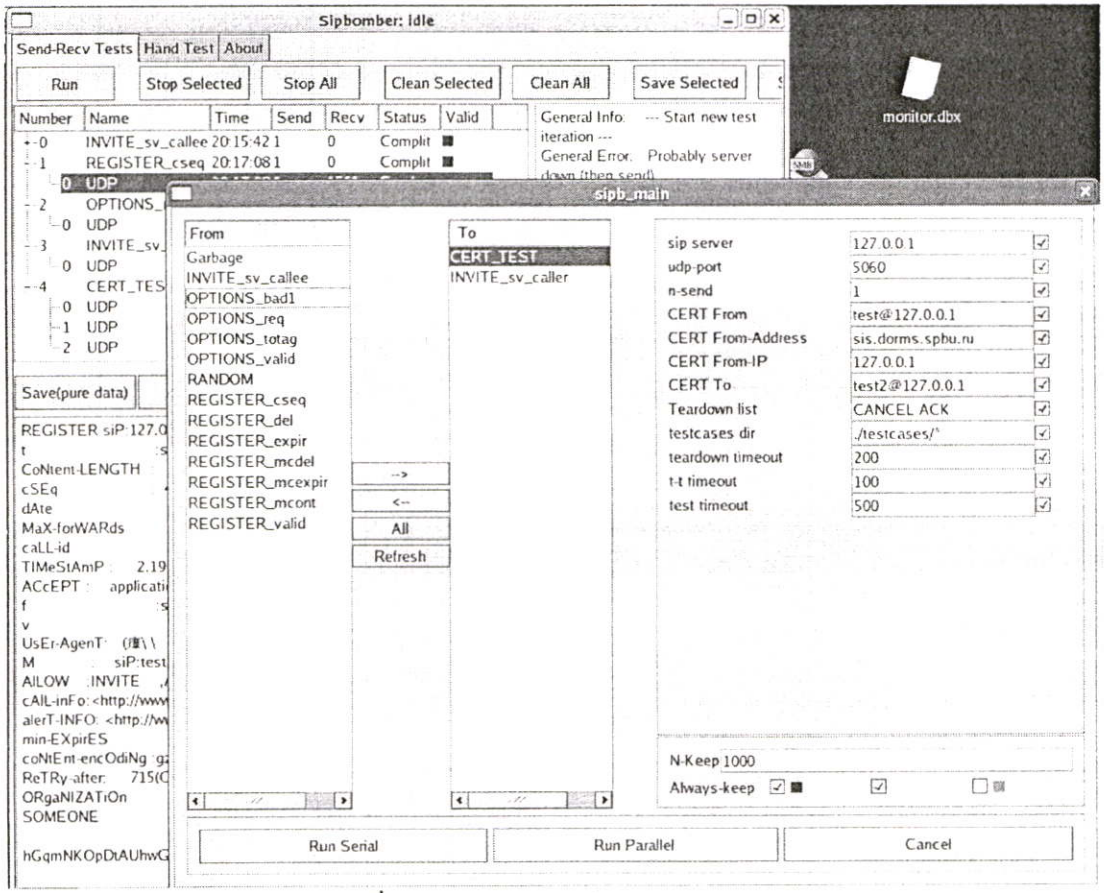
การทำงานของ Sipsak นั้นจะกระทำการผ่านทางคอมมานไลน์โดยจะต้องทำการเลือกสถานการณ์ที่มีอยู่แล้ว และทำการปรับเปลี่ยนฟิลด์บางฟิลด์ในเฮดเดอร์เพื่อทำการส่งต่อไป ฉะนั้นสถานการณ์ที่มีอยู่แล้วจึงเป็นตัวระบุความสามารถของ Sipsak คือไม่สามารถเพิ่มเติมหรือปรับเปลี่ยนสถานการณ์ตามแต่ที่ต้องการเองได้ เป็นเครื่องมือทดสอบที่ขาดความยืดหยุ่น ยกตัวอย่างเช่น การส่ง INVITE จะไม่สามารถปรับเปลี่ยนรูปแบบฟิลด์ในเฮดเดอร์ได้ รูปแบบการส่งเมสเสจในแต่ละขั้นตอนก็จะตายตัว

3.1.2 SipBomber 0.7

เป็นเครื่องมือที่ใช้ทดสอบซิพสำหรับลีนุกซ์ พัฒนาโดย Metalink ในปี 2003 ไว้ใช้เป็นการภายใน ซึ่งต่อมาได้ทำการเผยแพร่เป็นโอเพ่นซอร์ส GPL โดยการใช้ภาษา JAVA เป็นอินเตอร์เฟสและใช้ C/C++ เป็นแกนกลางในการทำงาน

SipBomber มีความสามารถดังนี้คือ

- สามารถส่ง OPTIONS request ไปยังเซิร์ฟเวอร์ชนิดใดๆได้
- REGISTER เพื่อให้ทดสอบการอิมพลีเมนต์ registrar server
- INVITE เพื่อให้ทดสอบซิพเซิร์ฟเวอร์ (registrar + proxy)
- CERT_TEST ใช้กับเซิร์ฟเวอร์ใดๆเพื่อทดสอบการทำงานกับ INVITE

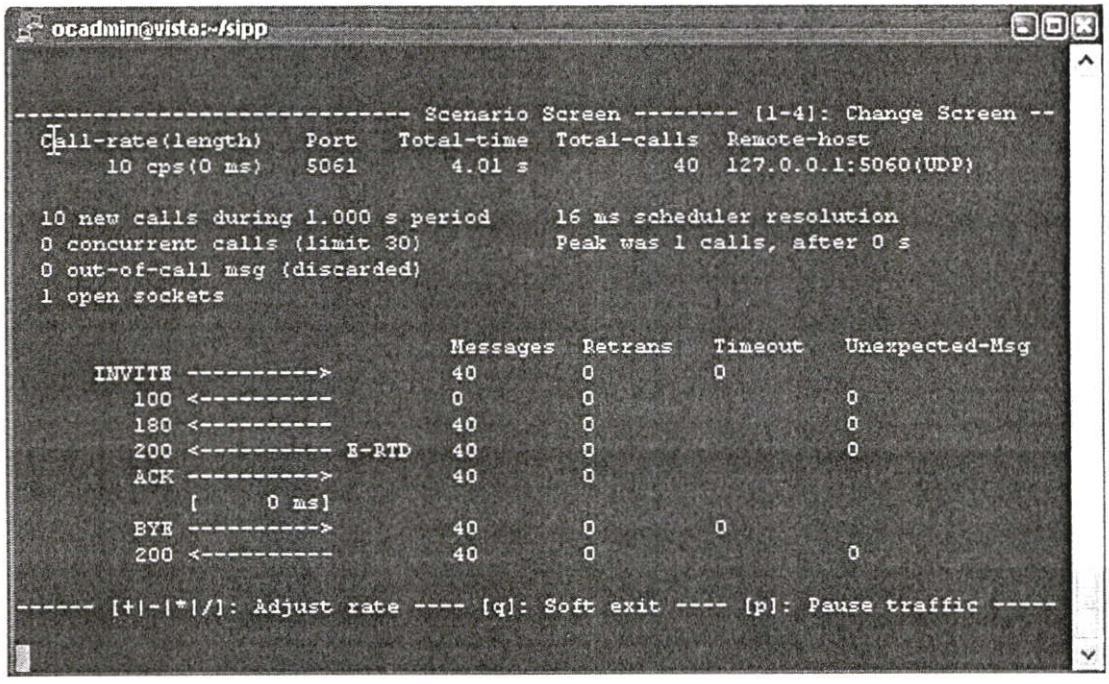


รูปที่ 3.3 ตัวอย่างการใช้งาน SipBomber

จะเห็นได้ว่า SipBomber ทำงานภายเมธอดได้แค่ 4 ชนิดเท่านั้น ซึ่งยังไม่เพียงพอต่อการทดสอบระบบ ถึงแม้ว่าจะมีการแสดงผลที่เข้าใจง่าย ผู้ใช้สามารถสร้างรูปแบบรีเควสขึ้นมาเองได้แต่ต้องภายใต้เมธอดที่กำหนดเท่านั้น แต่การตัดต่อสตริงไม่ว่าจะเป็นของรีเควสหรือเรสบอนส์ที่ถูกรับหรือจะส่งออกไปนั้นยังไม่ถูกต้อง

3.1.3 SIPp 1.0

เป็นเครื่องมือที่ใช้สร้างทราฟฟิคของซิพโปรโตคอล สามารถอ่านสถานการณ์ที่สร้างขึ้นเองได้โดยการใช้ไฟล์ XML ความสามารถของมันคือแสดงหน้าจอแบบไดนามิกเป็นสถิติที่ได้จากการรันทดสอบ (call rate, round trip delay และ message statistics) มีการทำรีทรานมิสชั่นและปรับเปลี่ยนอัตตราการเรียกสารได้ตลอดเวลา มันสามารถถูกใช้เพื่อทดสอบอุปกรณ์ซิพจริงๆได้หลายประเภท อย่างเช่น SIP proxies, B2BUAa, SIP media server, SIP/x gateways, SIP PBX มันจะมีประโยชน์อย่างมากในการจำลองการเรียกสายของ User Agent เป็นพันๆตัวในระบบซิพ



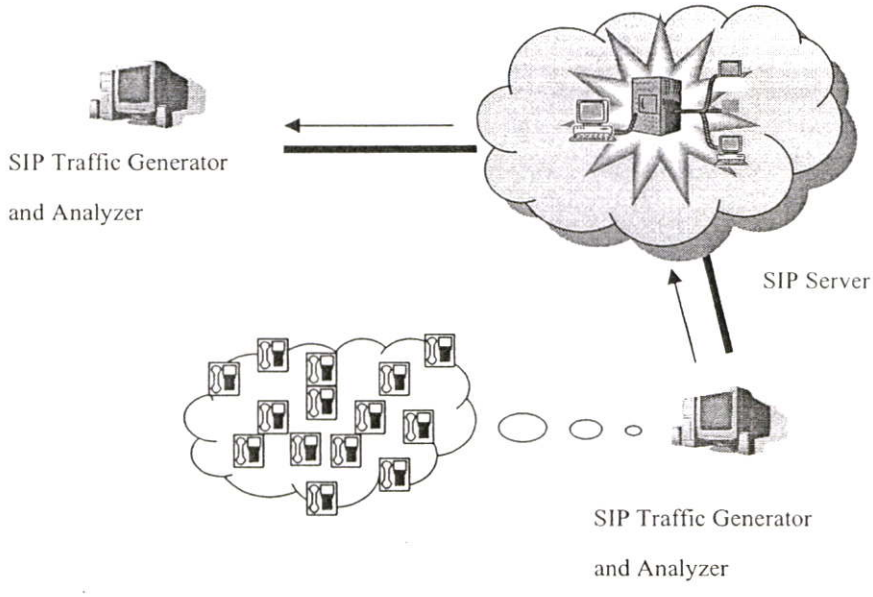
รูปที่ 3.4 ภาพการทำงานของ SIPp ในขณะทำการสร้าง INVITE

ฉะนั้นความสามารถของ SIPp จึงขึ้นอยู่การที่ผู้ใช้สามารถสร้างไฟล์สถานการณ์ XML ได้มากขนาดไหน จึงทำให้ผู้ใช้จะต้องเข้าใจถึงหลักการทำงานของระบบตนอย่างถูกต้อง และต้องอาศัยความรู้ในด้านของโปรโตคอลซิพอย่างมาก เพื่อให้ SIPp สามารถจำลองสถานการณ์ได้อย่างถูกต้องสมจริง และถึงแม้ว่า SIPp จะสามารถสร้างทราฟฟิกหรือเมสเสจได้ทีละจำนวนมากๆก็ตาม แต่ยังขาดการสนับสนุนการทำมัลติซิพยูซเซอร์ เนื่องจากทุกๆเมสเสจนั้นมาจาก XML เดียวกันทำให้ทุกๆการเรียกสายนั้นมีรูปแบบเดียวกันคือมาจากผู้ใช้เดียวกัน

3.2 คุณสมบัติของ SIP Traffic Generator and Analyzer

3.2.1 ลักษณะการทำงานทั่วๆไป

SIP Traffic Generator and Analyzer เป็นเครื่องมือที่ใช้ในการสร้างแพ็กเก็ตจำนวนมากของซิพ เสมือนเป็นการจำลองว่าเป็นการใช้งานจากอุปกรณ์ที่เป็นซิพจำนวนมาก เพื่อเป็นการทดสอบประสิทธิภาพของเซิร์ฟเวอร์หรือของเครือข่าย หรือเพื่อวัดประสิทธิภาพของอุปกรณ์ซิพเองก็ตาม และทำการวัดถึงผลลัพธ์ที่เกิดขึ้นจากสถานการณ์ที่สร้างขึ้นจาก SIP Traffic Generator and Analyzer ที่ใช้ในการทดสอบนั้นๆ เพื่อนำไปปรับปรุงอุปกรณ์ต่างๆ หรือเพื่อปรับแก้ถึงค่าพารามิเตอร์ที่ใช้ เช่น อายุของการทำ REGISTER แต่ละครั้งว่าควรจะเท่าไหร่ (ฟิลด์ Expires ในเฮดเดอร์) หรือผู้ใช้ควรจะทำการ REGISTER ทุกๆช่วงเวลาเท่าไหร่เพื่อให้เหมาะสมกับอายุของมัน และเพื่อวัดถึงดีเลย์หรืออัตราความล้มเหลวของแพ็กเก็ตที่เกิดขึ้นเพื่อกำหนดอายุที่เหมาะสมของการทำ REGISTER และกำหนดระยะเวลาของการทำรีทรานมิสชั่นของ REGISTER แพ็กเก็ตเมื่อกระทำการร้องขอไม่สำเร็จ



รูปที่ 3.5 การทำงานอย่างง่ายๆของ SIP Traffic Generator and Analyzer

จากรูปคือ SIP Traffic Generator and Analyzer ทั้ง 2 ตัว จะทำการโต้ตอบกันตามสถานการณ์ที่ได้กำหนดไว้ โดยทำการสื่อสารกันผ่านเครือข่ายต่างๆ โดยที่มีซิพเซิร์ฟเวอร์เป็นผู้ประสานงานการสื่อสารหรือเป็นผู้ควบคุม ทั้งนี้ SIP Traffic Generator and Analyzer จะจำลองตัวเองให้เป็นหลายๆผู้ใช้ที่ต่างกันและใช้งานในช่วงเวลาเดียวกันได้

3.2.2 เปรียบเทียบคุณสมบัติ

ได้ทำการเปรียบเทียบคุณสมบัติความสามารถระหว่าง SIP Traffic Generator and Analyzer กับซอฟต์แวร์อื่นๆที่มีอยู่ในปัจจุบัน เพื่อให้ทราบถึงคุณสมบัติเด่นของแต่ละซอฟต์แวร์ โดยซอฟต์แวร์ที่นำมาเปรียบเทียบมีดังนี้

1. SIPp 1.0 (SP)
2. Sipbomber 0.7 (SB)
3. Sipsak 0.8.12 (SS)
4. SIP Traffic Generator and Analyzer (Ours)

ตารางที่ 3.1 เปรียบเทียบคุณสมบัติกับซอฟต์แวร์อื่นๆ

คุณสมบัติและความสามารถ	SP	SB	SS	Ours
ภาษาที่พัฒนา	C/C++	Java + C/C++	C++	C/C++
แพลตฟอร์ม	Linux	Redhat9	Linux	Windows
สร้างแพ็กเก็ตจำนวนมากได้	✓	✓	✓	✓
ทดสอบระหว่าง UAS และ UAC	✓		✓	✓
สร้างรูปแบบสถานการณ์เองได้	✓	บางแพ็คเก็ต		✓
ส่งแพ็กเก็ตผ่านซิพเซิร์ฟเวอร์ต่างๆ	บางแพ็กเก็ต	บางแพ็กเก็ต	บางแพ็กเก็ต	บางแพ็กเก็ต
สร้างสถานการณ์เป็นมัลติยูซเซอร์				✓
โปรโตคอลในชั้นทรานสปอร์ต	TCP/UDP	TCP/UDP	UDP	TCP/UDP
GUI		✓		
คำนวณสถิติ	✓	✓		✓

3.3 การพัฒนาซอฟต์แวร์และลักษณะในการทำงาน

สำหรับการพัฒนานั้นได้นำซอฟต์แวร์ชื่อ SIPp เวอร์ชั่น 1.0 ซึ่งเป็นซอฟต์แวร์ที่เป็นโอเพ่นซอร์สที่เขียนด้วยภาษา C/C++ มา ซึ่งทำงานอยู่บนระบบปฏิบัติการลีนุกซ์ตระกูล Redhat โดยการทำงานจะอยู่บนเทอมินอล ซึ่งเราได้นำมาพอร์ทและแก้ไขเพิ่มเติมลงบน Microsoft Windows XP โดยใช้ Microsoft Visual C++ version 6 service pack 5 เป็นเครื่องมือ และให้มันสามารถทำงานบนคอมมานด์พร้อมท์บนวินโดวส์ได้

และทั้งนี้ SIPp นั้นเป็นซอฟต์แวร์โอเพ่นซอร์สที่นำเอาความสามารถพื้นฐานมาจากซอฟต์แวร์ชื่อ SIPSTONE [19] ที่พัฒนาโดย Columbia University และ Ubiquity ซึ่งเป็นผู้ริเริ่มและผลักดันให้โปรโตคอลซิพเกิดขึ้นมาและถูกใช้เป็นมาตรฐานขึ้น

โดยที่พวกเราได้ทำการเพิ่มความสามาถให้สามารถจำลองสถานการณ์เป็น Multiuser ให้มีความเหมือนกับการใช้งานจริง คือเหมือนกับมีหลายผู้ใช้ต่างๆกันเข้ามาใช้งานจำนวนมากแทนที่จะเป็นผู้ใช้เดียวกัน เนื่องจากหากเป็นการทำงานของผู้ใช้คนเดียวแล้วนั้นซิพเซิร์ฟเวอร์อาจจะทำการประมวลผลการร้องขอหรือตอบกลับเพียงครั้งเดียวเพราะเห็นว่าส่งมาจากหรือส่งไปยังผู้ใช้เดียวกัน ทั้งนี้ขึ้นอยู่กับโครงสร้างของซิพเซิร์ฟเวอร์แต่ละตัวด้วย ซึ่งหากจะทำการจำลองสถานการณ์โดยใช้เครื่องมือทดสอบหลายๆเครื่องก็เป็นไปได้ยากและไม่สะดวก และพวกเรายังปรับปรุงให้เป็นไปตามมาตรฐาน RFC3261 มากขึ้น รวมทั้งยังได้เพิ่มไฟล์สถานการณ์ที่เป็น XML ขึ้นในลักษณะการเรียกสายที่เป็นมัลติยูซเซอร์และให้ส่งรีเควสผ่าน SIP proxy แทนที่จะส่งระหว่าง UAC และ UAS โดยตรงเท่านั้น

3.3.1 การพอร์ทซอฟต์แวร์จากลีนุกซ์ขึ้นมาบนวินโดวส์

เราได้ทำการเปลี่ยนไลบรารี่ที่มีใช้งานอยู่เฉพาะบนลีนุกซ์ ให้เป็นไลบรารี่ที่มีอยู่บนวินโดวส์ คือ

1. ส่วนของ Socket เดิมบนลีนุกซ์มาเป็น winsock [20]

```
//#include <sys/socket.h>
 #include <winsock2.h>
```

2. ส่วนการทำงานเกี่ยวกับ regular expression ใช้ PCRE (Perl-compatible regular expression library) [21]

```
//#include <regex.h>
#include <pcreposix.h>
```

3. สำหรับคำสั่งการทำงานที่มีอยู่บนลีนุกซ์เท่านั้นจะใช้ UnixEm ทดแทน ซึ่งเป็น Synesis Software UNIX Emulation for Win32 [22]

และได้ทำการแก้ไขเพิ่มเติมซอร์สโค้ดที่ไม่สามารถหา ไลบรารี่มาทดแทนได้ โดยการเพิ่มฟังก์ชั่น, Struct หรือ Class เข้าไปทดแทน รวมทั้งทำการเปลี่ยนคำสั่งต่างๆให้เหมาะสมที่จะนำมาใช้บนวินโดวส์ เช่น ในไฟล์ sipp.cpp

```
int gettimeofday1(struct timeval *tv, void* tz)
{
        union
        {
                __int64 ns100; /*time since 1 Jan 1601 in 100ns units */
                FILETIME ft;
        } now;
```

```
        GetSystemTimeAsFileTime (&now.ft);
        tv->tv_usec = (long) ((now.ns100 / 10L) % 1000000L);
        tv->tv_sec = (long) ((now.ns100 - 116444736000000000L) / 10000000L);
        return (0);
}
```

3.3.2 การเพิ่มความสามารถให้เป็นมัลติยูซเซอร์

สาเหตุที่ต้องมีการปรับเปลี่ยนให้เป็นมัลติยูซเซอร์เนื่องจาก ต้องการให้ซอฟต์แวร์สามารถสร้างสถานการณ์ได้ใกล้เคียงหรือเหมือนกับสถานการณ์ที่สามารเกิดขึ้นจริงให้ได้มากที่สุด และจากบทที่ 2 เนื่องจากเซิร์ฟเวอร์ (Proxy, Redirect) นั้นมีการค้นหาในฐานข้อมูลหรือ service location ดังนั้นถ้ามีผู้ใช้หรือ User เข้ามาใช้งาน SIP URI เดียวกับหลายๆ SIP URI นั้นในด้านการค้นหาที่อยู่จึงอาจจะแตกต่างกันจากการค้นหาข้อมูล (จากเรคคอร์ดเดียวเรคคอร์ดเดิมในฐานข้อมูลกับหลายๆเรคคอร์ดที่แตกต่างกัน) ในจำนวนของการค้นหาที่เท่ากัน เช่นใน call.cpp

```
if (!next_user_number) {next_user_number++;}
new_call -> user_number = next_user_number;
...
if (next_user_number < max_user) next_user_number++;
else next_user_number = 1;
...
if(!next_user_number) { next_user_number ++; }
```

จึงได้ทำการปรับเปลี่ยน parser โดยการเพิ่มตัวแปรที่ชื่อ [user_id] ให้สามารถสร้างสถานการณ์ของ XML ให้ส่งเป็นมัลติยูซเซอร์ได้ ยกตัวอย่างเช่น

```
<scenario name="register">
<send retrans="500">
<![CDATA[
REGISTER sip:[remote_ip] SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port]
From: sip:UA[user_id]@[remote_ip]
To: sip:UA[user_id]@[remote_ip]
Call-ID: [call_id]
Cseq: 1 REGISTER
Contact: <sip:UA[user_id]@[local_ip]:[local_port]>
Expires: 3600
```

```
Content-Length: 0
]]>
</send>
</scenario>
```

คือเมื่อทำการส่งรีเควสที่เป็น REGISTER นี้ออกไปจะได้ผลลัพธ์ก็คือ จะเป็นการทำ REGISTER ในชื่อของ UA1@161.246.5.203 ถึง UA10@161.246.5.203 หากได้ทำการกำหนดให้ 161.246.5.203 เป็นแอดเดรสของเซิร์ฟเวอร์และมี max_user หรือจำนวนผู้ใช้เป็น 10

สรุปคือ การทำมัลติยูซเซอร์ของเราคือจะทำการแปลงหรือแปลจากไฟล์ XML ออกไป โดยจะส่งออกไปเรื่อยๆเป็นลำดับไปเรื่อยๆ จากผู้ใช้ที่หนึ่งถึงจำนวนผู้ใช้สูงสุดที่ได้กำหนดไว้ เมสเสจแต่ละอันก็จะส่งไปเป็นของแต่ละผู้ใช้ และเรียงกันออกไปเรื่อยๆ และเมื่อถึงจำนวนสูงสุดที่ได้กำหนดไว้ก็จะวนกลับมาเริ่มส่งตั้งแต่ผู้ใช้คนแรกหรือคนที่หนึ่งใหม่เริ่มต้นใหม่ไปเรื่อยๆจนกว่าจะกระทำการเสร็จตามที่กำหนดไว้

3.3.3 การเพิ่มความสามารถให้ส่งรีเควสส่งผ่านเซิร์ฟเวอร์ได้

จากความสามารถเดิมของ SIPp นั้นจะไม่สามารถกระทำบางเมธอดผ่านพร็อกซี่ได้ เนื่องจากมีการใช้ SIP URI เดียวเป็นตัวประสานงาน ซึ่งก็คือ AOI URI แต่ในความเป็นจริงแล้วยังจำเป็นจะต้องมี device URI ด้วย คือ SIPp จะใช้ [remote_address] เป็น AOI URI ในฟิลด์ภายในเฮดเดอร์ของเมธอดต่างๆ ยกตัวอย่างเช่น สำหรับซิพบางระบบนั้นจำเป็นจะต้องมีฟิลด์ ROUTE ในเฮดเดอร์ เพื่อเป็นตัวระบุเส้นทางในการส่งเมสเสจซึ่งในที่นี้ก็คือรีเควสและเรสบอนส์อย่าง VOCAL SIP server จะใช้ฟิลด์ ROUTE นี้ในขั้นตอน INVITE แต่เนื่องจาก SIPp ไม่รองรับการทำฟิลด์ ROUTE เพราะใช้ [remote_address] เดียวเท่านั้น ซึ่งจึงต้องเพิ่มตัวแปร [UAS_IP] ขึ้นมาเพื่อให้รองรับ URI ได้มากขึ้น ดังนั้นการเพิ่ม [UAS_IP] ขึ้นมาเป็น device URI จึงเป็นการเพิ่มประสิทธิภาพให้เป็นไปตามมาตรฐาน RFC3261 มากขึ้นด้วย เช่นใน sipp.cpp

```
if(!strcmp(argv[argi], "-UAS")) {
    if((++argi) < argc) {
      processed = 1;
      UAS_IP = argv[argi];
    } else {
      ERROR_P1("Missing argument for param '%s'.\n"
           "Use 'test -h' for details", argv[argi-1]);
    }
}
```

ตัวอย่างส่วนหนึ่งของไฟล์ XML ในขั้นตอนการ INVITE ที่จะส่ง ACK กลับไป

```
<send>
<![CDATA[
ACK sip:UA[user_id]@[remote_ip]:[remote_port] SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port]
From: <sip:UAC@[local_ip]:[local_port]>
To: <sip:UA[user_id]@[remote_ip]:[remote_port]>[peer_tag_param]
[last_Call-ID:]
Route: <sip:UA[user_id]@[remote_ip]:[remote_port];maddr=[remote_ip]>,<sip:UA[user_id]@[UAS_IP]:5060>
Cseq: 1 ACK
Content-Length: 0
]]>
</send>
```

ตัวอย่างนี้เป็นการสร้าง ACK response ของทาง UAC ผู้ที่ทำการ INVITE กลับไปยัง UAS เพื่อเป็นการยืนยัน 200 OK ที่ทาง UAS ส่งกลับมา

3.3.4 เพิ่มสูตรคำนวณประสิทธิภาพของระบบและเครือข่าย

ดำการเพิ่มสูตรการคำนวณนี้ลงใน erlang.hpp และ erlang.cpp

- Failed rate

คือ อัตราส่วนของการรีเควสที่ล้มเหลวจากการคีเควสทั้งหมด ซึ่งการรีเควสในที่นี้อาจจะหมายถึง REGISTER, INVITE หรือ SUBSCRIBE ก็แล้วแต่ขึ้นอยู่กับสถานการณ์ที่ใช้

$$Failed\ rate = \frac{Failed\ call}{Total\ call}\ x\ 100$$

- Offered load (Erlang A)

เป็นค่าที่บ่งบอกถึงปริมาณทราฟฟิกที่เกิดขึ้นและเข้ามาในระบบ มีหน่วยเป็น erlang

$$\boxed{a = \lambda\ T}$$

a = Erlang A

λ = arrival rate หรือ request / sec

T = holding time

```
double ErlangA(double arrival_rate,double holding_time)
{
        double ans = 0;
        //arrival rate = request per sec.
```

```
        return ans = arrival_rate * holding_time;
}
```

- Blocking probability, Probability of loss, Probability of rejection (Erlang B)

  คือ ความน่าจะเป็นที่รีเควส ที่เข้ามาจะไม่ประสบความสำเร็จหรือล้มเหลว

$$P(S) = \frac{\frac{a^S}{S!}}{\sum_{i=0}^{S} \frac{a^i}{i!}}$$

$P(S)$ = Erlang B

a = Erlang A จากที่ผ่านมา

S = จำนวนผู้ใช้สูงสุดเท่าที่ระบบจะรับได้

```
double ErlangB(double S,double a)
{
        double temp1 = 0;
        double temp2 = 0;
        double ans = 0;

        temp1 = (double)pow(a,S) / (double)fact(S);
        for (int i = 0 ; i <= S ; i++)
                temp2 += (double)pow(a,i) / (double)fact(i);

        return ans = temp1 / temp2;
}
```

- Probability of an arriving call being delayed (Erlang C)

  คือ ความน่าจะเป็นที่รีเควสที่เข้ามาจะหน่วงเวลาหรือถูกดีเลย์ไว้

$$C(S,a) = \frac{SB(S,a)}{S - a[1 - B(S,a)]} \quad for \quad a < S$$

$C(S,a)$ = Erlang C

$B(S,a)$ = Erlang B จากที่ผ่านมา

a = Erlang A จากที่ผ่านมา

S = จำนวนผู้ใช้สูงสุดเท่าที่ระบบจะรับได้

```
double ErlangC(double S,double a)
{
        //ErlangC = C(S,a) = Probability of an arriving call being delayed.
        double ans = 0;
        double b = ErlangB(S,a);

        return ans = (S * b) / (S - (a * (1 - b)));;
}
```

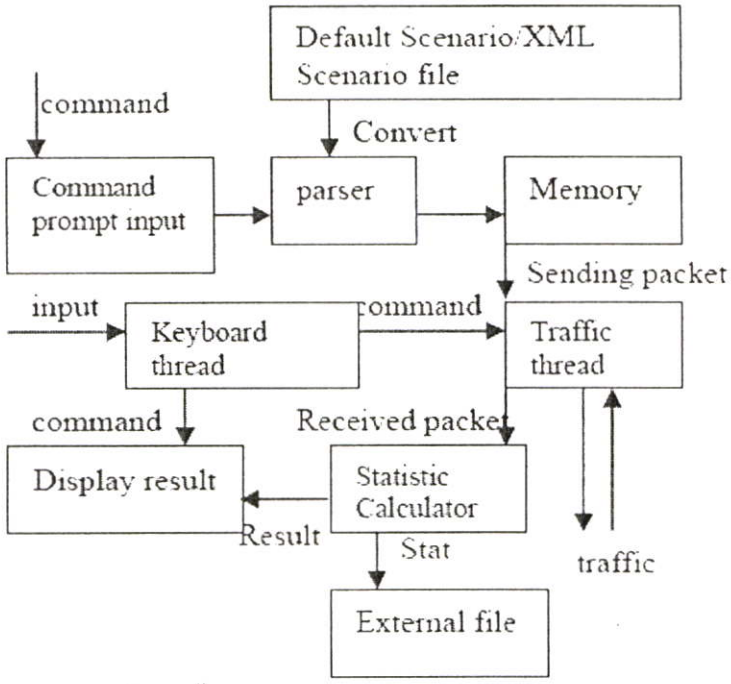
เนื่องจากการแสดงผลทางสถิตินั้นแบ่งเป็น 2 ส่วน คือ Periodic value และ Cumulative value ทำให้การคำนวณต่างๆจึงต้องแบ่งเป็น 2 ส่วนไปด้วย ซึ่ง

- Periodic value คือ ค่าที่เกิดจากภายในช่วงเวลาล่าสุด เช่น default จะเป็นค่าที่เกิดขึ้นภายในช่วงเวลา 1 วินาทีก่อนหน้านี้
- Cumulative value คือ ค่าสะสมสัมพัทธ์ตั้งแต่ได้เริ่มการทำงานของซอฟต์แวร์ขึ้นมาจนถึงปัจจุบันนี้ที่กำลังทำงานอยู่

Successful call	0	0
Failed call	0	54
Failed rate	0.000	0.000
Response Time	00:00:00:000	00:00:00:000
Call Length	00:00:00:000	00:00:00:000
S	10.000	10.000
Holding Time	5.000	5.000
ErlangA	0.000	5.886
ErlangB	0.000	0.040
ErlangC	0.000	0.091

รูปที่ 3.6 บางส่วนของหน้าจอการแสดงผลการคำนวณของซอฟต์แวร์

3.3.5 โครงสร้างที่สำคัญของซอฟต์แวร์



รูปที่ 3.7 ขั้นตอนการทำงานของซอฟต์แวร์

1. ส่วนรับคำสั่ง (sipp.cpp) – จะรับคำสั่งที่ต่อท้ายชื่อซอฟต์แวร์ทาง command prompt เช่น test –sf register.xml 161.246.5.203 หมายถึงทำการทดสอบโดยอ่านสถานการณ์จากไฟล์ XML ที่มีชื่อว่า register.xml และทำการส่งเมสเสจที่สร้างขึ้นนี้ไปยัง 161.246.5.203
2. ส่วน parser (scenario.cpp) – จะอ่านสถานการณ์จากไฟล์ XML หรือจาก default scenario เองที่มีอยู่ภายในซอฟต์แวร์มาเก็บไว้บนหน่วยความจำ เพื่อทำการรับส่งแพ็กเก็ตในลำดับต่อไป
3. ส่วน Keyboard thread (actions.cpp และ call.cpp) - เป็นส่วนที่รับอินพุททางคีย์บอร์ดในขณะที่ทำการรันซอฟต์แวร์อยู่เพื่อการแสดงผลที่ได้รับจากการทดสอบหรือเพื่อปรับแต่งอัตราการรับส่งต่างๆ ยกตัวอย่างเช่น กด + หรือ – ขณะซอฟต์แวร์กำลังทำงานเพื่อเพิ่มหรือลบอัตราการรีเควส
4. ส่วน Traffic thread (call.cpp) – เป็นส่วนที่ควบคุมการรับส่งแพ็กเก็ตทั้งหมดที่เป็นซิพรวมทั้งปรับเปลี่ยนการรับส่งให้เป็นไปตามสถานการณ์ที่สร้างขึ้น
5. ส่วนคำนวณสถิติ (stat.cpp) – จะทำการเก็บค่าหรือผลที่ได้รับจากการทดลองมาคำนวณแล้วเก็บไว้เพื่อแสดงผลที่หน้าจอต่อไป หรือทำการเก็บสถิติเพื่อลงไฟล์ และสามารถเก็บแพ็กเก็ตทั้งหมดที่เกิดขึ้นจากการรับส่งลงไฟล์ด้วย
6. ส่วนแสดงผล (screen.cpp) – แสดงผลที่ได้จากการคำนวณสถิติ หรือรูปแบบสถานการณ์ที่ใช้ทดสอบในขณะนั้นทางหน้าจอ

และอื่นๆอย่าง resource.c จัดการเกี่ยวกับทรัพยากรของเครื่อง, xp_parser ทำการดึงข้อมูลจากเครื่องที่เป็น WindowsXP และ variables.cpp เป็นประเภทตัวแปรที่ใช้ในโปรแกรม

บทที่ 4

การทดลองและผลการทดลอง

4.1 การทดลอง

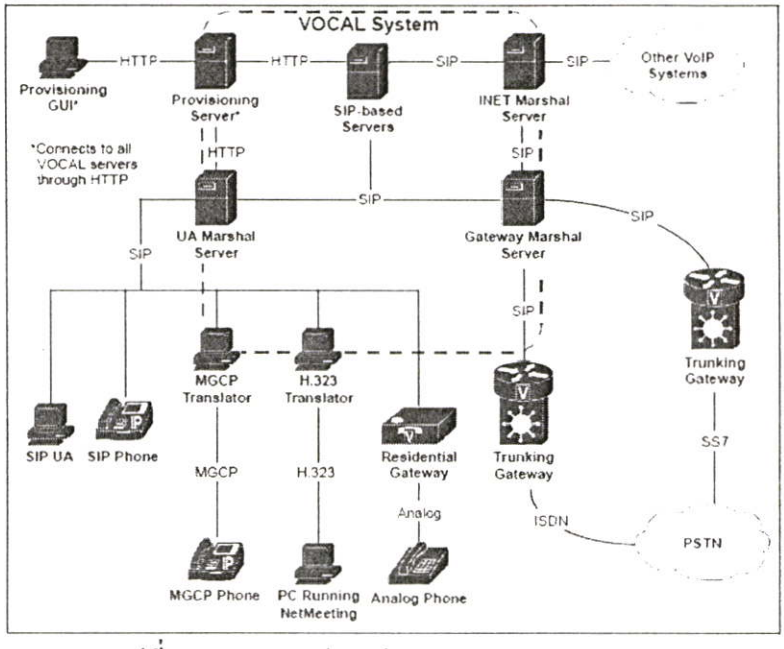
ในขั้นแรกเราได้ทดสอบความถูกต้องในการทำงานของโปรแกรมก่อน โดยการนำ SIP Traffic Generator and Analyzer และ SIPp ซึ่งก็คือซอฟต์แวร์ดั้งเดิมก่อนนำไปพัฒนามาทำการวัดประสิทธิภาพของ VOCAL ภายใต้สภาวะแวดล้อมและเงื่อนไขเดียวกัน เพื่อตรวจสอบความถูกต้องของซอฟต์แวร์หลังจากพัฒนาแล้ว

ต่อมานั้นเราได้นำซอฟต์แวร์ SIP Traffic Generator and Analyzer มาทดลองใช้โดยการนำไปทดสอบกับระบบซิพจำนวน 2 ระบบด้วยกัน ระบบแรกคือระบบที่เกิดจาก VOCAL SIP server [23] และอีกระบบคือ Partysip [24] ซึ่งทั้ง 2 ระบบนั้นจะมีลักษณะการทำงานและการใช้งานที่แตกต่างกันเกือบจะสิ้นเชิง เนื่องจากเราต้องการให้เห็นว่าซอฟต์แวร์ของเรานั้นมีความยืดหยุ่นในการใช้งาน สามารถนำไปทดสอบระบบใดๆก็ได้

โดยในการทดลองนั้น เราจะทดสอบประสิทธิภาพของระบบทั้งสองในการเรียกสายจากผู้ใช้กลุ่มหนึ่ง ไปยังผู้ใช้อีกกลุ่มหนึ่ง ดังนั้นเมธอดหลักๆที่จะใช้ในการทดสอบนี้คือ INVITE และอาจจะมีเมธอดอื่นๆเข้ามาร่วมเกี่ยวข้องเพื่อให้สามารถทำการ INVITE ได้อย่างสำเร็จและถูกต้อง (REGISTER, SUBSCRIBE และ NOTIFY)

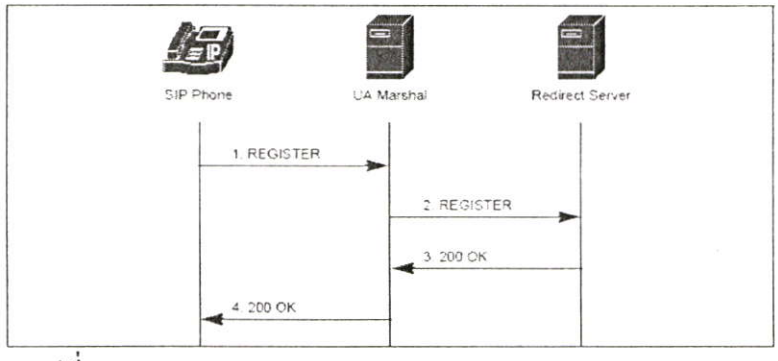
4.1.1 VOCAL SIP server

ระบบ VOCAL เป็น distributed network ของเหล่าเซิร์ฟเวอร์เพื่อให้บริการ Voice Over Internet Protocol (VoIP) รองรับอุปกรณ์การสื่อสารที่เป็น SIP (RFC3261), Media Gateway Control Protocol (MGCP) หรือ H.323 รวมทั้งยังรองรับโทรศัพท์อนาล็อกผ่าน resident gateway ด้วย



รูปที่ 4.1 มุมมองอย่างคร่าวๆของ VOCAL system

เพื่อจะให้ SIP phone สามารถเข้ามาใช้งานกับระบบได้ และเพื่อให้ SIP proxy หรือ SIP redirect server สามารถหาที่อยู่ของแต่ละคู่สื่อสารได้ จึงต้องมีการ REGISTER เกิดขึ้นก่อนอย่างอื่นทั้งหมด ซึ่งการ REGISTER ของ SIP phone เข้ากับ VOCAL system จะไม่ทำการรีเควสไปที่ registrar server แต่ว่าจะส่งไปที่ Redirect Server แทน โดยผ่านทาง Marshal Server (SIP proxy server) ดังรูป



รูปที่ 4.2 VOCAL : Call Flow Diagram : SIP phone registration

หาก SIP phone มีไอพีแอดเดรสเป็น 192.168.26.180, UA Marshal มีไอพีแอดเดรสเป็น 192.168.26.180, Redirect Server มีไอพีแอดเดรสเป็น 192.168.26.200 และต้องการใช้ AOI URI เป็น sip:6711@192.168.26.180

ฉะนั้นจึงมี REGISTER ที่ออกจาก SIP phone เป็นดังนี้คือ (1)

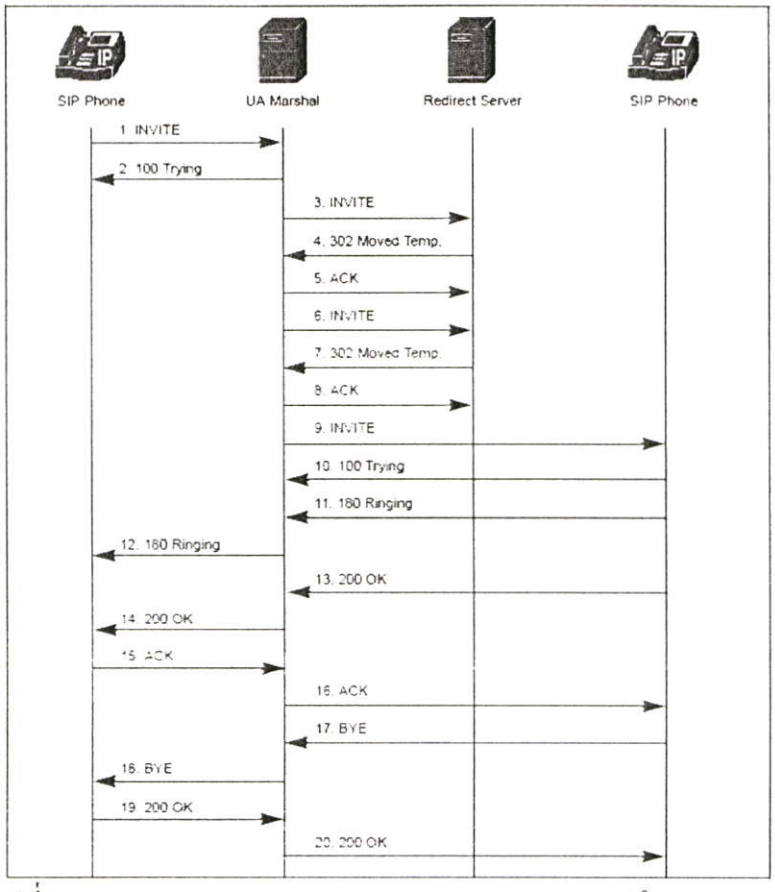
```
REGISTER sip:192.168.26.180 SIP/2.0 [192.168.26.10:50373->192.168.26.180:5060]
Via: SIP/2.0/UDP 192.168.26.10:5060
From: sip:6711@192.168.26.180
```

```
To: sip:6711@192.168.26.180
Call-ID: c2943000-1e262-14ae-2e323931@192.168.26.10
CSeq: 100 REGISTER
Contact: <sip:6711@192.168.26.10:5060>
Expires: 3600
Content-Length: 0
```

และมี ACK ที่กลับเข้ามายัง SIP phone เป็นดังนี้คือ (4)

```
SIP/2.0 200 OK [192.168.26.180:5060->192.168.26.10:5060]
Via: SIP/2.0/UDP 192.168.26.10:5060
From: <sip:6711@192.168.26.180:5060>
To: <sip:6711@192.168.26.180:5060>
Call-ID: c2943000-1e262-14ae-2e323931@192.168.26.10
CSeq: 100 REGISTER
Contact: <sip:6711@192.168.26.10:5060>
Expires: 3600
Content-Length: 0
```

การ INVITE ในรูปแบบที่เป็น SIP phone ทำการเรียกสายไปยังอีก SIP phone หนึ่งโดยผ่าน VOCAL system ซึ่งในที่นี้จะใช้ Marshal Server และ Redirect Server จะมีลักษณะดังรูปต่อไปนี้



รูปที่ 4.3 VOCAL : Call Flow Diagram : จาก SIP phone ถึง SIP phone

จากรูปหาก SIP phone ทางฝั่งซ้ายมี AOI URI เป็น sip:5121@192.168.36.180 และไอพีแอดเดรสเป็น 191.168.6.21 และอีกฝั่งเป็น sip:5120@192.168.36.180 และ 191.168.6.20 ตามลำดับ มี UA Mashal และ Redirect Server เป็น 192.168.36.180 และ 192.168.36.200 ตามลำดับแล้ว ดังนั้นซิพเมสเสจต่างๆที่ออกและเข้าทางฝั่งซ้ายดังนี้

เริ่มส่ง INVITE request ผ่านทาง UA Marshal (1)

```
INVITE sip:5120@192.168.36.180 SIP/2.0 [192.168.6.21:50623-
>192.168.36.180:5060]
Via: SIP/2.0/UDP 192.168.6.21:5060
From: sip:5121@192.168.6.21
To: <sip:5120@192.168.36.180>
Call-ID: c2943000-e0563-2a1ce-2e323931@192.168.6.21
CSeq: 100 INVITE
Expires: 180
User-Agent: Cisco IP Phone/ Rev. 1/ SIP enabled
Accept: application/sdp
Contact: sip:5121@192.168.6.21:5060
```

```
Content-Type: application/sdp
Content-Length: 219
v=0
o=CiscoSystemsSIP-IPPhone-UserAgent 16264 18299 IN IP4 192.168.6.21
s=SIP Call
c=IN IP4 192.168.6.21
t=0 0
m=audio 25282 RTP/AVP 0 101
a=rtpmap:0 pcmu/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-11
```

ได้รับ Provisioning response จาก UA Marshal (2)

```
SIP/2.0 100 Trying [192.168.36.180:5060->192.168.6.21:5060]
Via: SIP/2.0/UDP 192.168.6.21:5060
From: <sip:5121@192.168.6.21:5060>
To: <sip:5120@192.168.36.180:5060>
Call-ID: c2943000-e0563-2a1ce-2e323931@192.168.6.21
CSeq: 100 INVITE
Content-Length: 0
```

ได้รับ Provisioning response จาก UA Marshal (12)

```
SIP/2.0 180 Ringing [192.168.36.180:5060->192.168.6.21:5060]
Via: SIP/2.0/UDP 192.168.6.21:5060
From: <sip:5121@192.168.6.21:5060>
To: <sip:5120@192.168.36.180:5060>;tag=c29430002e0620-0
Call-ID: c2943000-e0563-2a1ce-2e323931@192.168.6.21
CSeq: 100 INVITE
Server: Cisco IP Phone/ Rev. 1/ SIP enabled
Content-Length: 0
```

ได้รับ Final response จาก UA Marshal (14)

```
SIP/2.0 200 OK [192.168.36.180:5060->192.168.6.21:5060]
Via: SIP/2.0/UDP 192.168.6.21:5060
```

```
From: <sip:5121@192.168.6.21:5060>
To: <sip:5120@192.168.36.180:5060>;tag=c29430002e0620-0
Call-ID: c2943000-e0563-2a1ce-2e323931@192.168.6.21
CSeq: 100 INVITE
Contact: <sip:5120@192.168.6.20:5060>
Record-Route:
<sip:5120@192.168.36.180:5060;maddr=192.168.36.180>,<sip:5120@192.168.36.180:5060;m
addr=1
92.168.36.180>
Cisco IP Phone/ Rev. 1/ SIP enabled
Content-Type: application/sdp
Content-Length: 218
v=0
o=CiscoSystemsSIP-IPPhone-UserAgent 13045 2886 IN IP4 192.168.6.20
s=SIP Call
c=IN IP4 192.168.6.20
t=0 0
m=audio 30658 RTP/AVP 0 101
a=rtpmap:0 pcmu/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-11
```

ส่ง ACK ของ INVITE ไปผ่านทาง UA Marshal (15)

```
ACK sip:5120@192.168.36.180:5060 SIP/2.0 [192.168.6.21:50623-
>192.168.36.180:5060]
Via: SIP/2.0/UDP 192.168.6.21:5060
From: <sip:5121@192.168.6.21:5060>
To: <sip:5120@192.168.36.180:5060>;tag=c29430002e0620-0
Call-ID: c2943000-e0563-2a1ce-2e323931@192.168.6.21
Route: <sip:5120@192.168.36.180:5060;maddr=192.168.36.180>,
<sip:5120@192.168.6.20:5060>
CSeq: 100 ACK
Content-Length: 0
```

ได้รับ BYE request ฝ่านมาจากทาง UA Mashal เพื่อเป็นการจัดเซสชั่น (18)

```
BYE sip:5121@192.168.6.21:5060 SIP/2.0 [192.168.36.180:5060-
>192.168.6.21:5060]
Via: SIP/2.0/UDP 192.168.36.180:5060;branch=4
Via: SIP/2.0/UDP 192.168.36.180:5060;branch=2
Via: SIP/2.0/UDP 192.168.6.20:5060
From: <sip:5120@192.168.36.180:5060>;tag=c29430002e0620-0
To: <sip:5121@192.168.6.21:5060>
Call-ID: c2943000-e0563-2a1ce-2e323931@192.168.6.21
CSeq: 101 BYE
Content-Length: 0
```

ส่ง Final response ของ BYE กลับไปทาง UA Marshal (19)

```
SIP/2.0 200 OK [192.168.6.21:50623->192.168.36.180:5060]
Via: SIP/2.0/UDP 192.168.36.180:5060;branch=4,SIP/2.0/UDP
192.168.36.180:5060;branch=2,SIP/2.0/UDP 192.168.6.20:5060
From: <sip:5120@192.168.36.180:5060>;tag=c29430002e0620-0
To: <sip:5121@192.168.6.21:5060>
Call-ID: c2943000-e0563-2a1ce-2e323931@192.168.6.21
Server: Cisco IP Phone/ Rev. 1/ SIP enabled
CSeq: 101 BYE
Content-Length: 0
```

และจากรูปที่ 4.3 นั้น ทาง SIP phone ทางฝั่งขวาก็จะได้รับและส่ง SIP message ต่างๆ ดังนี้

ได้รับ INVITE request ที่ถูกส่งผ่านมาทาง UA Marshal (9)

```
INVITE sip:5120@192.168.6.20:5060 SIP/2.0 [192.168.36.180:5060-
>192.168.6.20:5060]
Via: SIP/2.0/UDP 192.168.36.180:5060;branch=4
Via: SIP/2.0/UDP 192.168.36.180:5060;branch=2
Via: SIP/2.0/UDP 192.168.6.21:5060
From: <sip:5121@192.168.6.21:5060>
To: <sip:5120@192.168.36.180:5060>
Call-ID: c2943000-e0563-2a1ce-2e323931@192.168.6.21
```

```
CSeq: 100 INVITE
Expires: 180
Record-Route:
<sip:5120@192.168.36.180:5060;maddr=192.168.36.180>,<sip:5120@192.168.36.180:5060;m
addr=1
92.168.36.180>
Contact: <sip:5121@192.168.6.21:5060>
Content-Type: application/sdp
Content-Length: 219
v=0
o=CiscoSystemsSIP-IPPhone-UserAgent 16264 18299 IN IP4 192.168.6.21
s=SIP Call
c=IN IP4 192.168.6.21
t=0 0
m=audio 25282 RTP/AVP 0 101
a=rtpmap:0 pcmu/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-11
```

ส่ง provisioning response กลับไป UA Marshal (10)

```
SIP/2.0 100 Trying [192.168.6.20:50753->192.168.36.180:5060]
Via: SIP/2.0/UDP 192.168.36.180:5060;branch=4,SIP/2.0/UDP
192.168.36.180:5060;branch=2,SIP/2.0/UDP 192.168.6.21:5060
From: <sip:5121@192.168.6.21:5060>
To: <sip:5120@192.168.36.180:5060>;tag=c29430002e0620-0
Call-ID: c2943000-e0563-2a1ce-2e323931@192.168.6.21
Server: Cisco IP Phone/ Rev. 1/ SIP enabled
CSeq: 100 INVITE
Content-Length: 0
```

ส่ง provisioning response กลับไป UA Marshal (11)

```
SIP/2.0 180 Ringing [192.168.6.20:50753->192.168.36.180:5060]
Via: SIP/2.0/UDP 192.168.36.180:5060;branch=4,SIP/2.0/UDP
192.168.36.180:5060;branch=2,SIP/2.0/UDP 192.168.6.21:5060
```

```
From: <sip:5121@192.168.6.21:5060>
To: <sip:5120@192.168.36.180:5060>;tag=c29430002e0620-0
Call-ID: c2943000-e0563-2a1ce-2e323931@192.168.6.21
Server: Cisco IP Phone/ Rev. 1/ SIP enabled
CSeq: 100 INVITE
Content-Length: 0
```

ส่ง final response กลับไป UA Marshal เพื่อเป็นการบอกว่ายอมรับ INVITE แล้ว (13)

```
SIP/2.0 200 OK [192.168.6.20:50753->192.168.36.180:5060]
Via: SIP/2.0/UDP 192.168.36.180:5060;branch=4,SIP/2.0/UDP
192.168.36.180:5060;branch=2,SIP/2.0/UDP 192.168.6.21:5060
From: <sip:5121@192.168.6.21:5060>
To: <sip:5120@192.168.36.180:5060>;tag=c29430002e0620-0
Call-ID: c2943000-e0563-2a1ce-2e323931@192.168.6.21
Server: Cisco IP Phone/ Rev. 1/ SIP enabled
Contact: sip:5120@192.168.6.20:5060
Record-Route:
<sip:5120@192.168.36.180:5060;maddr=192.168.36.180>,<sip:5120@192.168.36.180:5060;m
addr=1
92.168.36.180>
CSeq: 100 INVITE
Content-Type: application/sdp
Content-Length: 218
v=0
o=CiscoSystemsSIP-IPPhone-UserAgent 13045 2886 IN IP4 192.168.6.20
s=SIP Call
c=IN IP4 192.168.6.20
t=0 0
m=audio 30658 RTP/AVP 0 101
a=rtpmap:0 pcmu/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-11
```

ได้รับ ACK ที่ถูกส่งผ่านมาทาง UA Marshal เป็นการบ่งบอกให้เริ่มเซสชั่น (16)

```
ACK sip:5120@192.168.6.20:5060 SIP/2.0 [192.168.36.180:5060-
>192.168.6.20:5060]
Via: SIP/2.0/UDP 192.168.36.180:5060;branch=4
Via: SIP/2.0/UDP 192.168.36.180:5060;branch=2
Via: SIP/2.0/UDP 192.168.6.21:5060
From: <sip:5121@192.168.6.21:5060>
To: <sip:5120@192.168.36.180:5060>;tag=c29430002e0620-0
Call-ID: c2943000-e0563-2a1ce-2e323931@192.168.6.21
CSeq: 100 ACK
Content-Length: 0
```

ส่ง BYE request กลับไป โดยผ่านทาง UA Marshal เพื่อเป็นการบอกให้ตัดเซสชั่น (17)

```
BYE sip:5120@192.168.36.180:5060 SIP/2.0 [192.168.6.20:50753-
>192.168.36.180:5060]
Via: SIP/2.0/UDP 192.168.6.20:5060
From: <sip:5120@192.168.36.180:5060>;tag=c29430002e0620-0
To: <sip:5121@192.168.6.21:5060>
Call-ID: c2943000-e0563-2a1ce-2e323931@192.168.6.21
User-Agent: Cisco IP Phone/ Rev. 1/ SIP enabled
CSeq: 101 BYE
Route: <sip:5120@192.168.36.180:5060;maddr=192.168.36.180>,
<sip:5121@192.168.6.21:5060>
Content-Length: 0
```

ได้รับ final response ที่ถูกส่งกลับผ่านมาทาง UA Marshal (20)

```
SIP/2.0 200 OK [192.168.36.180:5060->192.168.6.20:5060]
Via: SIP/2.0/UDP 192.168.6.20:5060
From: <sip:5120@192.168.36.180:5060>;tag=c29430002e0620-0
To: <sip:5121@192.168.6.21:5060>
Call-ID: c2943000-e0563-2a1ce-2e323931@192.168.6.21
CSeq: 101 BYE
Server: Cisco IP Phone/ Rev. 1/ SIP enabled
Content-Length: 0
```

จากเมสเสจทั้งหมดที่ถูกรับและส่งไปมาระหว่าง SIP phone นั้น จะสังเกตได้ว่าสำหรับ VOCAL system ในการทำ REGISTER, INVITE และ BYE นั้น ไม่มีซิพเมสเสจใดเลยที่ไม่ผ่านทาง UA Mashal จึงเป็นข้อแตกต่างจากบทที่ 2 แต่ก็ไม่ได้ผิดหลักการแต่อย่างใด และการสร้างและตัดเซสชั่นนั้นก็ยังเป็นไปได้ด้วยดี

4.1.2 Partysip

Partysip นั้นเป็น SIP proxy server ซึ่งทำงานได้ตามมาตรฐาน RFC2543 โดยที่ Partysip นั้นเป็นโมดูลล่าแอพพลิเคชั่นซึ่งก็คือมีโมดูลการทำงานแบ่งเป็นส่วนๆประกอบเข้าด้วยกัน ซึ่งเราสามารถเพิ่มหรือลดความสามารถของ Partysip ได้โดยการเพิ่มหรือลดปลั๊กอินของมัน และในขณะนี้ Partysip เมื่อร่วมกันกับปลั๊กอินของมันทำให้มันสามารถถูกใช้เป็น SIP registrar, SIP redirect server และ SIP statefull proxy server ได้

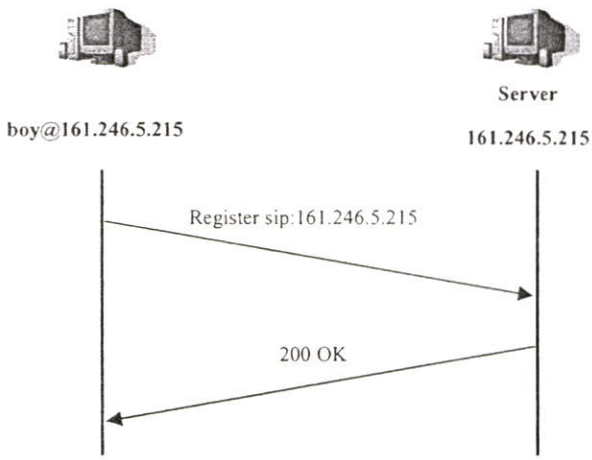
Partysip ใช้ libosip2 ในการทำงาน โดยที่ใช้ libosip2 ทำงานเป็นเหมือนกับ parser ให้กับส่วนของซิพเมสเสจเพื่อเป็นตัวบ่งบอกว่าซิพเมสเสจนี้เป็นชนิดใด และมีในแต่ละส่วนของซิพเมสเสจบ่งบอกถึงอะไรบ้าง

ส่วนประกอบของ Partysip และปลั๊กอินของมันมีดังนี้ คือ

1. Partysip – เป็นส่วนหลักของทั้งหมด มีหน้าที่คือควบคุมการทำงานของระบบ และทำหน้าที่เรียกใช้งานปลั๊กอินอื่นๆให้สามารถนำมาทำงานร่วมกันได้
2. ppl – เป็นไลบรารี่ที่ใช้ในการติดต่อกับระบบปฏิบัติการหรือโอเอสเพื่อใช้เรียกข้อมูลในส่วนต่างๆ ยกตัวอย่างเช่น time, socket, dns, md5
3. psp_auth – ทำหน้าที่ในการ Authenticate ให้กับผู้ใช้ที่ทำการส่ง REGISTER request ให้กับระบบ
4. psp_filter – ทำหน้าที่ในส่วนของ statefull proxy server ทำหน้าที่กรองซิพเมสเสจที่มีเงื่อนไขเป็นไปตามที่กำหนด
5. psp_ls_localdb – ทำหน้าที่ในส่วนของ redirect server เก็บข้อมูลในส่วนของ record-routing ซึ่งเป็นส่วนหนึ่งในซิพเมสเสจ เมื่อซิพเซิร์ฟเวอร์ทำงานในโหมดที่เป็น statefull proxy และทำหน้าที่เป็นเสมือนกับฐานข้อมูลเพื่อบ่งบอกว่าผู้ใช้นี้อยู่ในตำแหน่งที่สถานที่ใด (ไอพีแอดเดรส) และทำการ forking ให้กับเมสเสจ
6. psp_ls_sfull – ทำหน้าที่ในส่วนของ redirect server เก็บข้อมูลในส่วนของ record-routing ซึ่งเป็นส่วนหนึ่งในซิพเมสเสจ
7. psp_ls_static – ทำหน้าที่เป็นเหมือนกับ redirect server ทำการปฏิเสธซิพเมสเสจที่มี URI ที่ไม่รู้จัก และส่งต่อซิพเมสเสจ
8. psp_rgstrar – ทำหน้าที่รับ REGISTER request ของผู้ใช้เพื่อทำการ registration ผู้ใช้นั้นๆ เข้ากับระบบ

9. psp_syntax – ทำหน้าที่ในการตรวจสอบไวยกรณ์ให้กับซิพเมสเสจที่เข้ามาในระบบ
10. psp_udp – ทำหน้าที่ในการรับและส่งซิพเมสเสจ

และเช่นเดียวกันนั้น สำหรับ Partysip เริ่มแรกผู้ใช้ก็จะต้องทำการ REGISTER เข้ากับระบบเสียก่อน ซึ่งจะส่ง REGISTER request ไปยัง Partysip เลยเนื่องจากว่า Partysip เป็น SIP application server ที่ประกอบจากโมดูลต่างๆ ดังนั้นทั้ง Registrar, Proxy และ Redirect server จึงทำงานอยู่บนเครื่องและบนแอพพลิเคชั่นเดียวกันเลย



รูปที่ 4.4 การ REGISTER สำหรับ Partysip

หาก UA มีไอพีแอดเดรสเป็น 161.246.5.209 ต้องการใช้ AOI URI เป็น sip:boy@161.246.5.215 และสำหรับ Partysip มีไอพีแอดเดรสเป็น 161.246.5.215

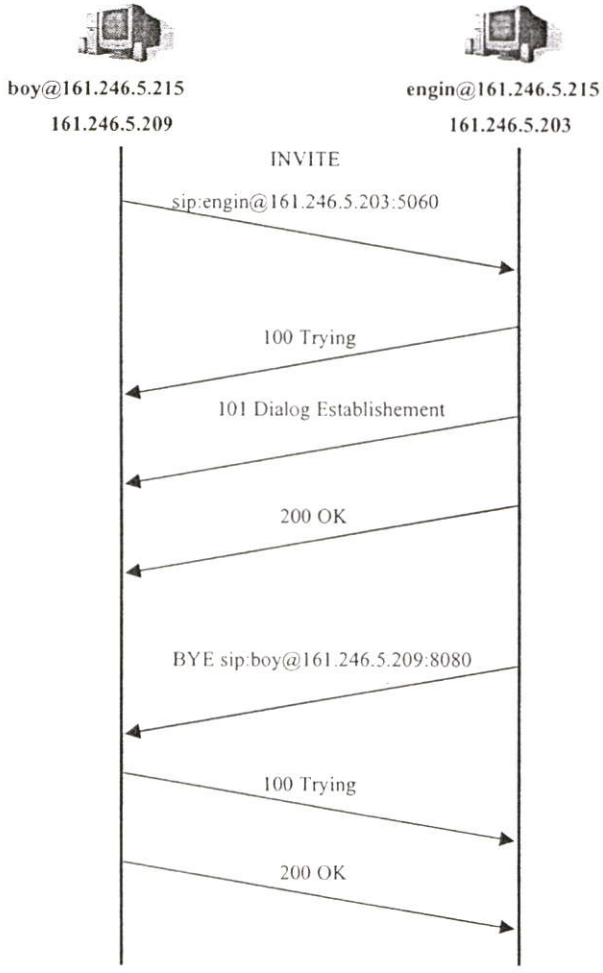
ฉะนั้นจึงมี REGISTER ที่ออกจาก UA เป็นดังนี้คือ

```
REGISTER sip:161.246.5.215 SIP/2.0
Via: SIP/2.0/UDP 161.246.5.209:5060;rport;branch=z9hG4bk28870
From: <sip:boy@161.246.5.215>;tag=17649
To: <sip:boy@161.246.5.215>
Call-ID: 25514@161.246.5.209
CSeq: 1 REGISTER
Contact: <sip:161.246.5.209:5060>
Max-Forwards: 5
Expires: 3600
Content-Length: 0
```

และเมื่อขั้นตอนการ REGISTER สำหรับเรียบร้อยแล้ว ทาง Partysip ก็จะส่ง final response (200 OK) กลับดังนี้

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP 161.246.5.209:5060;rport;branch=z9hG4bk28870
From: <sip:boy@161.246.5.215>;tag=17649
To: <sip:boy@161.246.5.215>tag:15724
Call-ID: 25514@161.246.5.209
CSeq: 1 REGISTER
Contact: <sip:161.246.5.209:5060>
Expires: 3600
Content-Length: 0
```

หลังจากเสร็จสิ้นจึงเริ่มการ INVITE ได้ โดยมีรูปแบบเป็นดังรูป

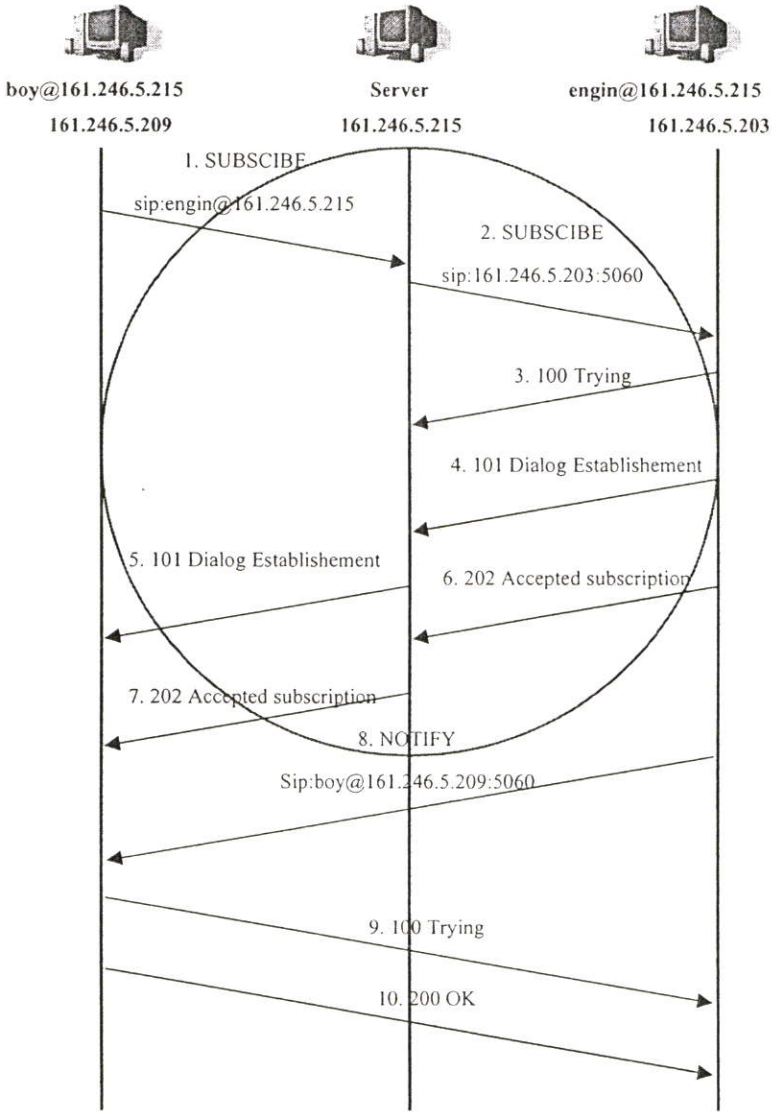


รูปที่ 4.5 การ INVITE สำหรับ Partysip

จากรูปจะสังเกตได้ว่า สำหรับการทำ INVITE นี้แล้วจะไม่มีเมสเสจใดๆเลยที่ส่งผ่าน Partysip แต่ UA ทราบถึงที่อยู่ของอีกฝั่งหนึ่งได้อย่างไร? ก็เพราะว่าสำหรับ Partysip นั้นจะมีการทำ SUBSCRIBE และ NOTIFY ระหว่าง UAs ตลอดเวลาเพื่อให้ UAs แต่ละตัวทราบถึงที่อยู่ของ

กันและกันได้ ดังนั้นการทำ NVITE ของ Partysip จึงไม่เป็นโหลดให้กับระบบเซิร์ฟเวอร์เลย แต่จะไปเป็นโหลดให้กับการทำ SUBSCRIBE และ NOTIFY แทน ดังนั้นจึงข้อความรูปแบบของเมสเสจที่เป็น INVITE ทั้งหมดไป

สำหรับการทำ SUBSCRIBE และ NOTIFY ของ Partysip นั้นจะมีรูปแบบดังรูปต่อไปนี้



รูปที่ 4.6 ขั้นตอนการทำ SUBSCRIBE และ NOTIFY

จากรูปที่ 4.6 จะเห็นได้ว่ามีเพียงแค่ SUBSCRIBE เท่านั้นที่เป็นโหลดให้กับเซิร์ฟเวอร์ ดังนั้นจึงขอแสดงเมสเสจเฉพาะในส่วนที่วงกลมสีแดงไว้เท่านั้น

ซิพเมสเสจของ UA ทางฝั่งซ้ายมือ ซึ่งก็คือ boy@161.246.5.215 ที่มีไอพีแอดเดรสเป็น 161.246.5.209 ทั้งที่เข้าและออกจะมีรูปแบบดังนี้

SUBSCRIBE แรกเพื่อถามไปยัง engin ว่ามีสถานะเป็นอย่างไร (1)

```
SUBSCRIBE sip:engin@161.246.5.215 SIP/2.0
Via: SIP/2.0/UDP 161.246.5.209:5060;rport;branch=z9hG4bk26500
From: <sip:boy@161.246.5.215>;tag=18467
```

```
To: <sip:engin@161.246.5.215>
Call-ID: 6334@161.246.5.215
CSeq: 20 SUBSCRIBE
Contact: <sip:boy@161.246.5.209:5060>
Max-Forwards: 5
Event: presence
User-Agent: exosip/0.1
Expires: 3600
Allow: INVITE, ACK, CANCEL, BYE, OPTIONS, REFER, SUBSCRIBE, NOTIFY, MESSAGE
Accept: application/pidf+xml
Content-Length: 0
```

ซึ่งจะได้รับ provisioning response กลับมา โดยผ่านกลับมาทาง Partysip (5)

```
SIP/2.0 101 Dialog Establishement
Via: SIP/2.0/UDP 161.246.5.209:5060;rport=5060;branch=z9hG4bk26500
From: <sip:boy@161.246.5.215>;tag=18467
To: <sip:engine@161.246.5.215>;tag=41
Call-ID: 6334@161.246.5.215
CSeq: 20 SUBSCRIBE
Contact: <sip:engin@161.246.5.203:5060>
Event: presence
Allow: INVITE, ACK, CANCEL, BYE, OPTIONS, REFER, SUBSCRIBE, NOTIFY, MESSAGE
Content-Length: 0
```

ต่อไปเมื่อทาง engin ยอมรับการทำ SUBSCRIBE นั้น boy จะได้รับ final response กลับมา (7)

```
SIP/2.0 202 Accepted subscription
Via: SIP/2.0/UDP 161.246.5.209:5060;rport=5060;branch=z9hG4bk26500
From: <sip:boy@161.246.5.215>;tag=18467
To: <sip:engin@161.246.5.215>;tag=41
Call-ID: 6334@161.246.5.215
CSeq: 20 SUBSCRIBE
```

```
Contact: <sip:engin@161.246.5.209:5060>
Event: presence
Expires: 3600
Allow: INVITE, ACK, CANCEL, BYE, OPTIONS, REFER, SUBSCRIBE, NOTIFY,
MESSAGE
Content-Length: 0
```

และสำหรบ UA อีกฝั่งหนึ่ง ซึ่งก็คือ engin@161.246.5.215 ที่มี IP address เป็น 161.246.5.203 จะมี SIP message ทั้งที่เข้าและออกจะมีรูปแบบดังนี้

เริ่มแรกด้วยการที่ UA ได้รับ SUBSCRIBE จากอีก UA หนึ่งที่ผ่านมาทาง Partysip (2)

```
SUBSCRIBE sip:161.246.5.203:5060 SIP/2.0
Via: SIP/2.0/UDP 161.246.5.215:5060;branch=z9hG4bkf3f1db8750029057b6fa5e6b3f6d338
40.0
Via: SIP/2.0/UDP 161.246.5.209:5060;rport;branch=z9hG4bk26500
From: <sip:boy@161.246.5.215>;tag=18467
To: <sip:engin@161.246.5.215>
Call-ID: 6334@161.246.5.209
CSeq: 20 SUBSCRIBE
Contact: <sip:boy@161.246.5.209:5060>
Max-Forwards: 4
Event: presence
User-Agent: exosip/0.1
Expires: 3600
Allow: INVITE, ACK, CANCEL, BYE, OPTIONS, REFER, SUBSCRIBE, NOTIFY,
MESSAGE
Accept: application/pidf+xml
Content-Length: 0
```

เมื่อได้รับ SUBSCRIBE แล้ว ก็จะทำการส่ง provisioning response กลับไปก่อน ซึ่งก็คือ 100 Trying และ 101 Dialog Estrablishement (3), (4)

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP 161.246.5.215:5060;branch=z9hG4bkf3f1db8750029057b6fa5e6b3f6d338
40.0
Via: SIP/2.0/UDP 161.246.5.209:5060;rport;branch=z9hG4bk26500
```

From: <sip:boy@161.246.5.215>;tag=18467

To: <sip:engin@161.246.5.215>

Call-ID: 6334@161.246.5.209

CSeq: 20 SUBSCRIBE

Event: presence

Allow: INVITE, ACK, CANCEL, BYE, OPTIONS, REFER, SUBSCRIBE, NOTIFY, MESSAGE

Content-Length: 0

SIP/2.0 101 Dialog Establishement

Via: SIP/2.0/UDP 161.246.5.215:5060;branch=z9hG4bkf3f1db8750029057b6fa5e6b3f6d338 40.0

Via: SIP/2.0/UDP 161.246.5.209:5060;rport;branch=z9hG4bk26500

From: <sip:boy@161.246.5.215>;tag=18467

To: <sip:engin@161.246.5.215>;tag=41

Call-ID: 6334@161.246.5.209

CSeq: 20 SUBSCRIBE

Event: presence

Allow: INVITE, ACK, CANCEL, BYE, OPTIONS, REFER, SUBSCRIBE, NOTIFY, MESSAGE

Content-Length: 0

และสุดท้ายของการ SUBSCRIBE คือ ยอมรับและส่ง final response กลับไปยังผู้ request โดยผ่านทาง Partysip (6)

SIP/2.0 202 Accepted subscription

Via: SIP/2.0/UDP 161.246.5.215:5060;branch=z9hG4bkf3f1db8750029057b6fa5e6b3f6d338 40.0

Via: SIP/2.0/UDP 161.246.5.209:5060;rport;branch=z9hG4bk26500

From: <sip:boy@161.246.5.215>;tag=18467

To: <sip:engin@161.246.5.215>;tag=41

Call-ID: 6334@161.246.5.209

CSeq: 20 SUBSCRIBE

Event: presence

Expires: 3600

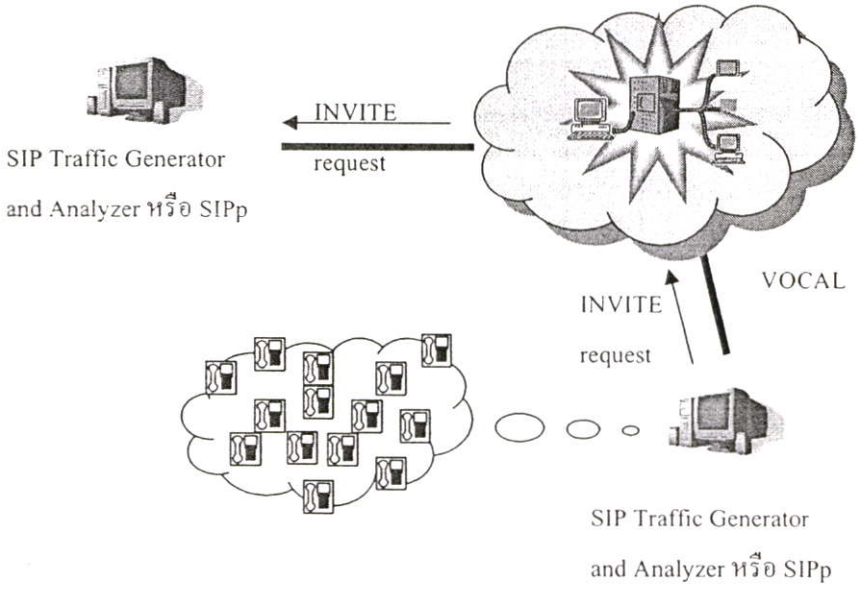
Allow: INVITE, ACK, CANCEL, BYE, OPTIONS, REFER, SUBSCRIBE, NOTIFY, MESSAGE

Content-Length: 0

4.2 ผลการทดลอง

4.2.1 SIPp กับ SIP Traffic Generator and Analyzer

ได้ทำการสร้างสถานการณ์ที่เป็นการ INVITE และ BYE โดยใช้ SIPp และ SIP Traffic Generator and Analyzer โดยจะเป็นการทำรีเควสจากยูซเซอร์หรือผู้ใช้ที่ชื่อ UAC ไปยังอีกผู้หนึ่งชื่อ UAS โดยการใช้งานจะเป็นดังรูปต่อไปนี้

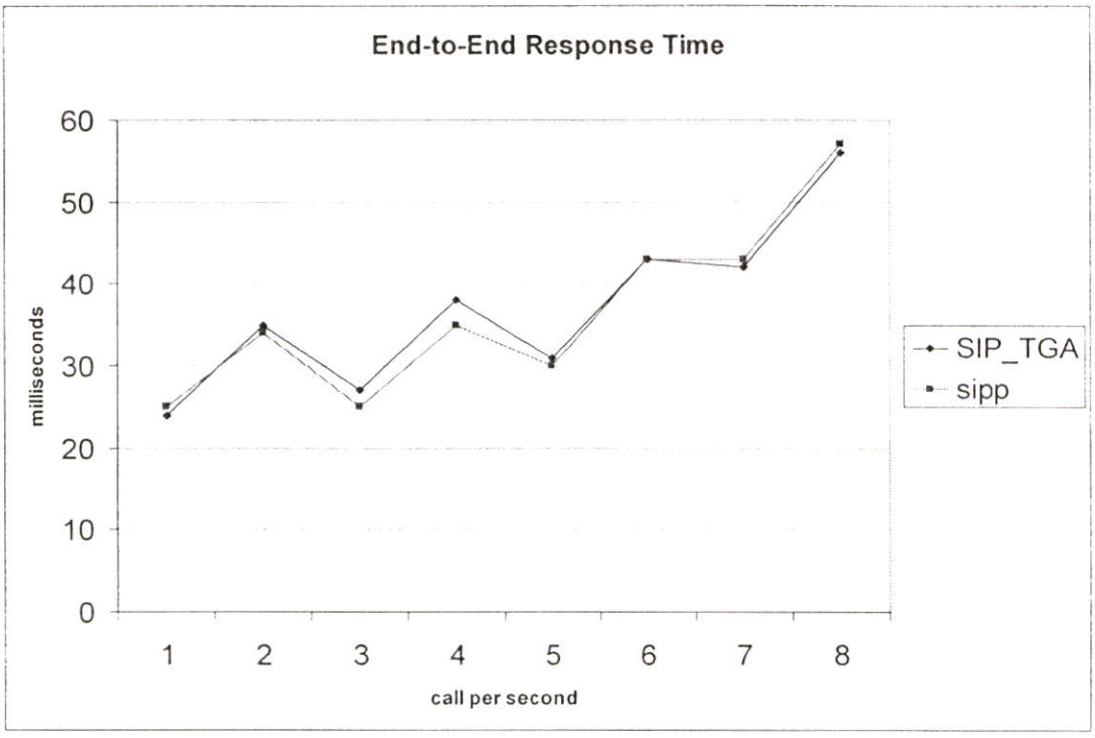


รูปที่ 4.7 วิธีการใช้ SIPp และ SIP Traffic Generator and Analyzer เพื่อสร้าง INVITE และวัดผล

ทั้งนี้ใน VOCAL นั้นมีผู้ใช้ทั้งหมด 50 คน ซึ่งจะได้ผลลัพธ์ดังตารางต่อไปนี้

ตารางที่ 4.1 เปรียบเทียบผลลัพธ์การใช้งานของ SIPp และ SIP Traffic Generator and Analyzer

	SIPp	**SIP Traffic Generator and Analyzer**
Request rate (call per sec)	**Response Time (ms)**	**Response Time (ms)**
1	24	25
2	35	34
3	27	25
4	38	35
5	31	30
6	43	43
7	42	43
8	56	57



รูปที่ 4.8กราฟแสดง End-to-End average response time (ms) ที่วัดได้จาก SIPp และ SIP Traffic Generator and Analyzer

ซึ่งจะเห็นได้ว่าจากการใช้งานใช้งานข้างต้นหลังจากที่ได้ทำการปรับปรุงและเปลี่ยนแปลงให้กับ SIPp จนกลายเป็น SIP Traffic Generator แล้วก็คงผลลัพธ์ถูกต้องเดิมไว้ไม่เปลี่ยนแปลง

การทดลองในส่วนต่อไปเราได้ใช้ SIP Traffic Generator and Analyzer เพื่อวัดประสิทธิภาพของ SIP server ทั้งสองประเภท คือ VOCAL และ Partysip

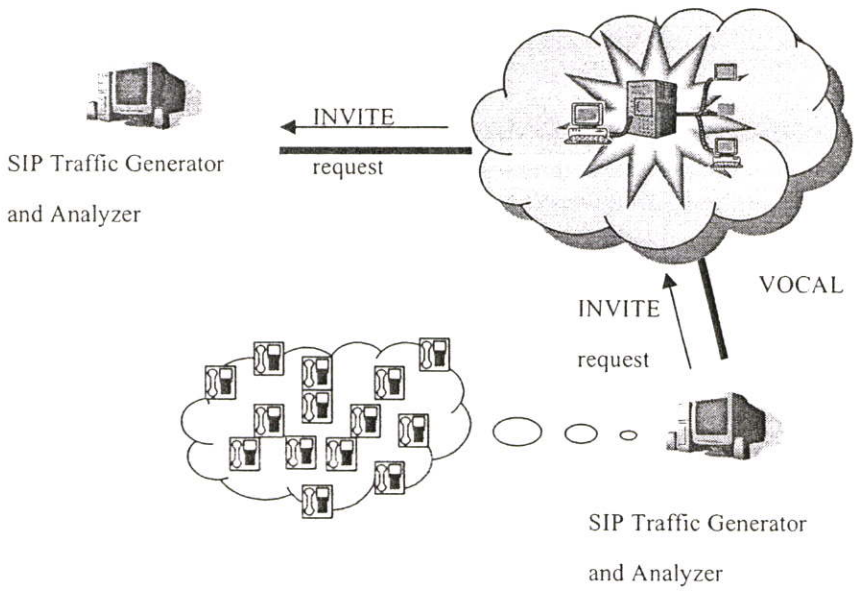
ตารางที่ 4.2 VOCAL vs Partysip environment

	VOCAL	**Partysip**
OS	Linux Redhat 9.0	Windows XP service pack 2
CPU	Intel Celeron 2.2 GHz	Intel Celeron 2.2 GHz
Memory	256 MB / DDR266	256 MB / DDR266
Interface	1 LAN card 100 Mb/sec	1 LAN card 100 Mb/sec

4.2.2 VOCAL SIP server

รันระบบ VOCAL ทั้งหมด โดยให้เซิร์ฟเวอร์ทั้งหมด (Marshal, Redirect และอื่นๆ) ทำงานอยู่บนเครื่องคอมพิวเตอร์เครื่องเดียวกัน

ในการทดลองนั้นจะใช้สถานการณ์ตามรูปที่ 4.2 และรูปที่ 4.3 ตามลำดับ โดยอ่านจากไฟล์ XML ที่สร้างไว้เพื่อให้มีเมสเสจตรงตามที่ได้กล่าวไว้ข้างต้น และการใช้งานซอฟต์แวร์ SIP Traffic Generator and Analyzer ก็จะเป็นดังรูป



รูปที่ 4.9 วิธีการใช้ SIP Traffic Generator and Analyzer เพื่อสร้าง INVITE และวัดผล

- ใช้ VOCAL_M_REGISTER.xml ในการทำ REGISTER ให้กับ SIP Traffic Generator and Analyzer โดยให้ฝั่งที่ทำหน้าที่เป็น UAC ใช้ชื่อ sip:UAC และให้ฝั่งที่เป็น UAS ใช้ชื่อ sip:UAxx (xx คือหมายเลขของผู้ใช้ เนื่องจากจะให้ UASs มีหลายๆที่ ยกตัวอย่างเช่น UA1, UA2, ...)

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<scenario name="register">
<send retrans="500">
<![CDATA[
REGISTER sip:[remote_ip] SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port]
From: sip:UA[user_id]@[remote_ip]
To: sip:UA[user_id]@[remote_ip]
Call-ID: [call_id]
Cseq: 1 REGISTER
Contact: <sip:UA[user_id]@[local_ip]:[local_port]>
Expires: 3600
Content-Length: 0
```

```
]]>
</send>
<recv response="100" optional="true">
</recv>
<recv response="400" optional="true">
</recv>
<recv response="401" optional="true">
</recv>
<recv response="402" optional="true">
</recv>
<recv response="403" optional="true">
</recv>
<recv response="200" crlf="true">
</recv>
</scenario>
```

หลังจากทำการ REGISTER เรียบร้อยหมดแล้ว จึงทำการ INVITE จาก UAC (sip:UAC) ไปยัง UAS (sip:UAxx) ทุกๆตัว โดยใช้

- VOCAL_M_UAC_INVITE.xml ในฝั่งที่เป็น UAC เพื่อสร้าง INVITE

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<scenario name="UAC invite">
<send retrans="500">
<![CDATA[
INVITE sip:UA[user_id]@[remote_ip] SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port]
From: sip:UAC@[local_ip]
To: <sip:UA[user_id]@[remote_ip]>
Call-ID: [call_id]
Cseq: 2 INVITE
Expires: 180
User-Agent: Cisco IP Phone/ Rev. 1/ SIP enabled
Accept: aplication/sdp
Contact: sip:UAC@[local_ip]:[local_port]
```

```
Content-Type: application/sdp
Content-Length: 221
v=0
o=CiscoSystemsSIP-IPPhone-UserAgent 16264 18299 IN IP4 [local_ip]
s=SIP Call
c=IN IP4 [local_ip]
t=0 0
m=audio 25282 RTP/AVP 0 101
a=rtpmap:0 pcmu/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-11
]]>
</send>
<recv response="100" optional="true">
</recv>
<recv response="180" optional="true">
</recv>
<recv response="200" rtd="true">
</recv>
<send>
<![CDATA[
ACK sip:UA[user_id]@[remote_ip]:[remote_port] SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port]
From: <sip:UAC@[local_ip]:[local_port]>
To: <sip:UA[user_id]@[remote_ip]:[remote_port]>[peer_tag_param]
[last_Call-ID:]
Route:
<sip:UA[user_id]@[remote_ip]:[remote_port];maddr=[remote_ip]>,<sip:UA[user_id]@[UAS_I
P]:5060>
Cseq: 2 ACK
Content-Length: 0
]]>
</send>
```

```
<pause milliseconds="5000"/>
<send retrans="500">
<![CDATA[
BYE sip:UAC@[remote_ip]:[remote_port] SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port]
From: <sip:UAC@[remote_ip]:[remote_port]>;tag=[call_number]
To: <sip:UA[user_id]@[UAS_IP]>
[last_Call-ID:]
User-Agent: Cisco IP Phone/ Rev. 1/ SIP enabled
Cseg: 3 BYE
Route:
<sip:UAC@[remote_ip]:[remote_port];maddr=[remote_ip]>,<sip:UA[user_id]@[UAS_IP]:506
0>
Content-Length: 0
]]>
</send>
<recv response="200" crlf="true">
</recv>
<ResponseTimeRepartition value="10, 20, 30, 40, 50, 100, 150, 200"/>
<CallLengthRepartition value="5000,6000,7000,8000,9000,10000"/>
</scenario>
```

- VOCAL_M_UAS_INVITE.xml ในฝั่งที่เป็น UAS เพื่อรับ INVITE โต้ตอบกัน

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<scenario name="UAS invite">
<recv request="INVITE" crlf="true">
</recv>
<send>
<![CDATA[
SIP/2.0 100 Trying
[last_Via:]
[last_From:]
[last_To:];tag=[call_number]
```

```
[last_Call-ID:]
Server: Cisco IP Phone/ Rev. 1/SIP enabled
[last_CSeq:]
Content-Length: 0
]]>
</send>
<send>
<![CDATA[
SIP/2.0 180 Ringing
[last_Via:]
[last_From:]
[last_To:];tag=[call_number]
[last_Call-ID:]
Server: Cisco IP Phone/ Rev. 1/SIP enabled
[last_CSeq:]
Content-Length: 0
]]>
</send>
<send retrans="500">
<![CDATA[
SIP/2.0 200 OK
[last_Via:]
[last_From:]
[last_To:];tag=[call_number]
[last_Call-ID:]
Server: Cisco IP Phone/Rev. 1/SIP enabled
Contact: sip:uas@[local_ip]:[local_port]
[last_Record-Route]
[last_CSeq:]
Content-Type: application/sdp
Content-Length: 220
v=0
o=CiscoSystemsSIP-IPPhone-UserAgent 13045 2886 IN IP4 [local_ip]
```

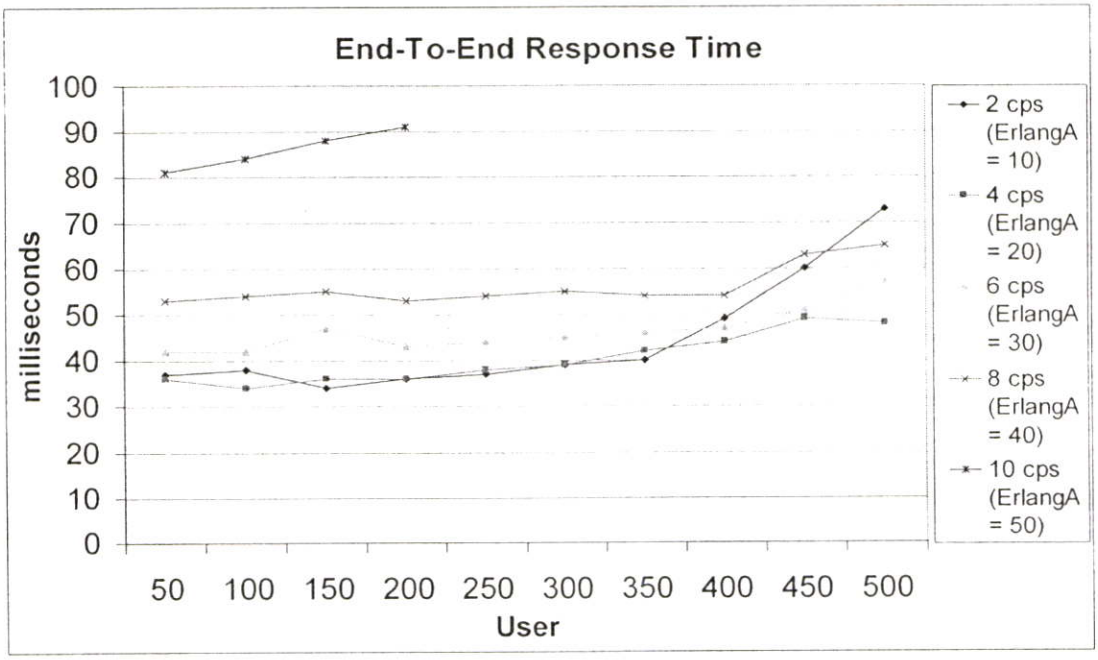
```
s=SIP Call
c=IN IP4 [local_ip]
t=0 0
m=audio 30658 RTP/AVP 0 101
a=rtpmap:0 pcmu/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-11
]]>
</send>
<recv request="ACK" rtd="true" crlf="true">
</recv>
<recv request="BYE">
</recv>
<send>
<![CDATA[
SIP/2.0 200 OK
[last_Via:]
[last_From:][peer_tag_param]
[last_To:]
[last_Call-ID:]
Server: Cisco IP Phone/ Rev. 1/ SIP enabled
[last_CSeq:]
Content-Length: 0
]]>
</send>
</scenario>
```

เมื่อทำการวัดประสิทธิภาพของ VOCAL จะได้ผลลัพธ์รูปของ

- Failure rate (%) คือ อัตราส่วนของการทำ INVITE จนถึง BYE ไม่สำเร็จ
- End-to-End average response time (ms) คือ ค่าเฉลี่ยของระยะเวลาที่ใช้ไปเมื่อเริ่มทำการส่ง INVITE จนกระทั่งได้รับ 100 Trying กลับมา
- Blocking Probability (%) คือ ความน่าจะเป็นที่ INVITE นั้นๆจะไม่สำเร็จต่อ Offer Load ที่เข้ามาในระบบ

ตารางที่ 4.3 ผลลัพธ์ของ Average Response Time ของ VOCAL

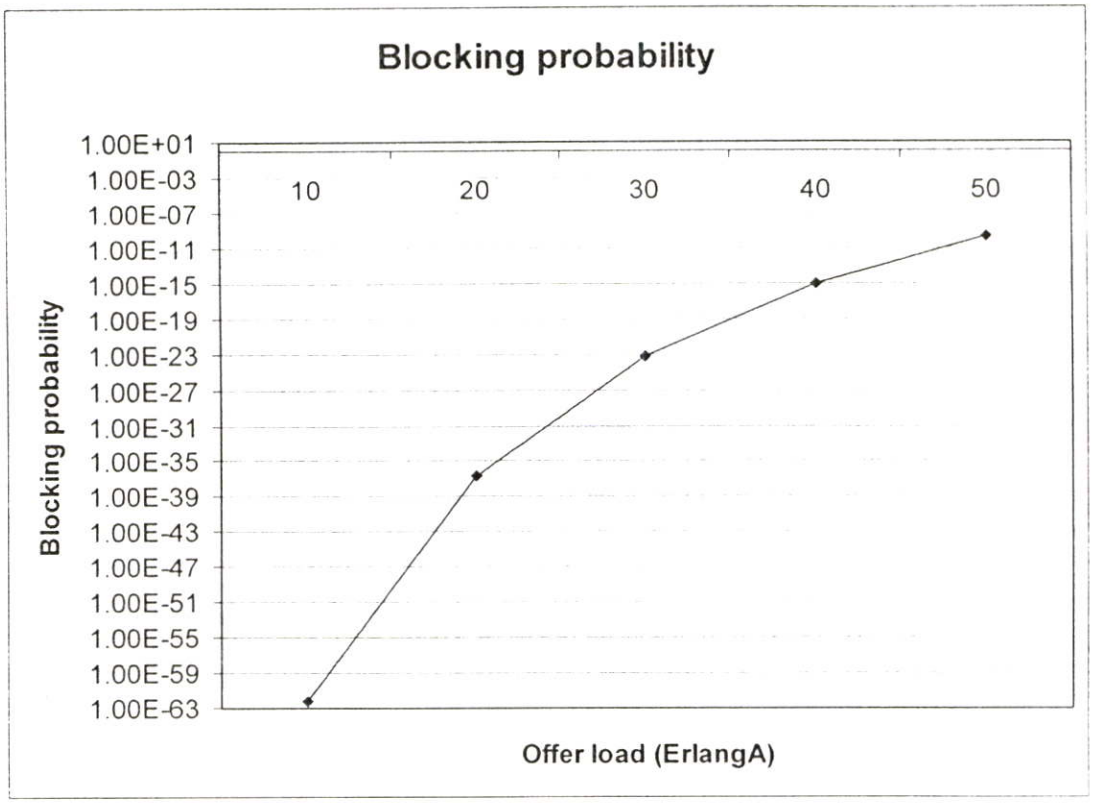
	2 cps (ErlangA = 10)	4 cps (ErlangA = 20)	6 cps (ErlangA = 30)	8 cps (ErlangA = 40)	10 cps (ErlangA = 50)
User	ResponseTime (ms)	ResponseTime (ms)	ResponseTime (ms)	ResponseTime (ms)	ResponseTime (ms)
50	37	36	42	53	81
100	38	34	42	54	84
150	34	36	47	55	88
200	36	36	43	53	91
250	37	38	44	54	-
300	39	39	45	55	-
350	40	42	46	54	-
400	49	44	47	54	-
450	60	49	51	63	-
500	73	48	57	65	-



รูปที่ 4.10 กราฟแสดง End-to-End average response time (ms) ของ VOCAL (INVITE)

ตารางที่ 4.4 ผลลัพธ์การระหว่าง Offer Load (Erlang A) และ Blocking Probability ของ VOCAL

	Blocking Probability				
Users	ErlangA=10	ErlangA=20	ErlangA=30	ErlangA=40	ErlangA=50
100	4.865E-63	2.799E-37	5.168E-24	7.32E-16	1.63E-10



รูปที่ 4.11 กราฟแสดง Offer Load และ Blocking Probability ของ VOCAL (INVITE) ในขณะที่มีจำนวนผู้ใช้งาน 100 ยูซเซอร์

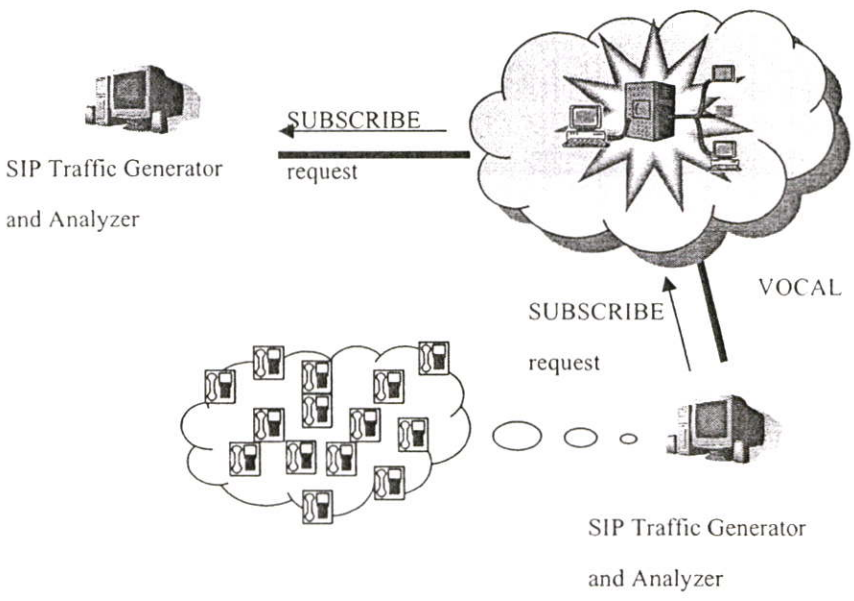
และผลของการวัด Failure rate ได้ว่าเป็น 0.000% เนื่องจากหากเริ่มมี Failure rate เกิดขึ้นแล้ว VOCAL จะเกิดอาการแฮงค์และทำให้ End-to-End average response time (ms) พุ่งสูงขึ้นเรื่อยจนกระทั่ง VOCAL ดับไป

สรุปผลการทดลองของ VOCAL จากกราฟจะได้ว่า ไม่สามารถที่จะรับ INVITE ใดๆได้เกิน 10 request/sec เนื่องจากจะทำให้เซิร์ฟเวอร์ไม่เสถียร และจำนวนผู้ใช้ที่เหมาะสมคือไม่เกิน 400 คน เพราะเกินจากนี้ความชันของกราฟจะเริ่มพุ่งขึ้นอย่างรวดเร็ว และ End-to-End average response time นั้นมีค่าอยู่ไม่เกิน 91 ms ฉะนั้นจึงควรตั้งเวลาในการทำ retransmission ของ INVITE ให้มากกว่า 91 ms และจำนวนผู้ใช้ทีผลน้อยมากหากเทียบกับจำนวน INVITE ที่เข้ามาในระบบ

4.2.3 Partysip

รันระบบ Partysip โดยเปิดใช้ทุกโมดูลของมัน (Register, Proxy และอื่นๆ) บนเครื่องคอมพิวเตอร์หนึ่งเครื่อง

โดยที่ในการทดลองนั้นจะใช้สถานการณ์ตามรูปที่ 4.4 และรูปที่ 4.6 (เฉพาะส่วนที่เกี่ยวข้องกับเซิร์ฟเวอร์ในส่วนวงกลมสีแดง) ตามลำดับ เนื่องจากรูปที่ 4.5 ไม่มีส่วนเกี่ยวข้องกับเซิร์ฟเวอร์เลย โดยอ่านจากไฟล์ XML ที่สร้างไว้เพื่อให้มีเมสเสจตรงตามที่ได้กล่าวไว้ข้างต้น และการใช้งานซอฟต์แวร์ SIP Traffic Generator and Analyzer ก็จะเป็นดังรูป



รูปที่ 4.12 วิธีการใช้ SIP Traffic Generator and Analyzer เพื่อสร้าง SUBSCRIBE และวัดผล

- ใช้ Partysip_M_REGISTER.xml ในการทำ REGISTER ให้กับ SIP Traffic Generator and Analyzer โดยให้ฝั่งที่ทำหน้าที่เป็น UAC ใช้ชื่อ sip:UAC และให้ฝั่งที่เป็น UAS ใช้ชื่อ sip:UAxx (xx คือหมายเลขของผู้ใช้ เนื่องจากจะให้ UASs มีหลายๆที่ ยกตัวอย่างเช่น UA1, UA2, ...)

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<scenario name="register">
<send retrans="500">
<![CDATA[
REGISTER sip:[remote_ip] SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port];rport=[remote_port]
From: <sip:UA[user_id]@[remote_ip]>;tag=[call_number]
To: <sip:UA[user_id]@[remote_ip]>
Call-ID: [call_id]
Cseq: 1 REGISTER
Contact: <sip:[local_ip]:[local_port]>
Max-Forwards: 5
User-Agent: exosip/0.1
Expires: 3600
Content-Length: 0
]]>
```

```
</send>
<recv response="100" optional="true">
</recv>
<recv response="400" optional="true">
</recv>
<recv response="401" optional="true">
</recv>
<recv response="402" optional="true">
</recv>
<recv response="403" optional="true">
</recv>
<recv response="200" crlf="true">
</recv>
</scenario>
```

หลังจากทำการ REGISTER เรียบร้อยหมดแล้ว จึงทำการ SUBSCRIBE และ จาก UAC (sip:UAC) ไปยัง UAS (sip:UAxx) ทุกๆตัว โดยใช้

- Partysip_M_SUBSCRIBE.xml ในฝั่งที่เป็น UAC เพื่อส่ง SUBSCRIBE request ออกไป

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<scenario name="UAC subscribe">
<send retrans="500">
<![CDATA[
SUBSCRIBE sip:UA[user_id]@[remote_ip] SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port];rport=[remote_port]
From: <sip:UAC@[remote_ip]>;tag=[call_number]
To: <sip:UA[user_id]@[remote_ip]>
Call-ID: [call_id]
Cseq: 20 SUBSCRIBE
Contact: <sip:UAC@[local_ip]:[local_port]>
Max-Forwards: 5
Event: presence
User-Agent: exosip/0.1
Expires: 3600
```

```
Allow: INVITE, ACK, CANCEL, BYE, OPTIONS, REFER, SUBSCRIBE, NOTIFY,
MESSAGE
Accept: application/pidf+xml
Content-Length: 0
]]>
</send>
<recv response="480" optional="true" rtd="true" crlf="true">
</recv>
<recv response="101" optional="true">
</recv>
<recv response="202" rtd="true" crlf="true">
</recv>
<ResponseTimeRepartition value="10, 20, 30, 40, 50, 100, 150, 200"/>
<CallLengthRepartition value="5000,6000,7000,8000,9000,10000"/>
</scenario>
```

- Partysip_M_NOTIFY.xml ในฝั่งที่เป็น UAS เพื่อรับ SUBSCRIBE และโต้ตอบกลับ

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<scenario name="UAC subscribe">
<recv request="SUBSCRIBE" crlf="true">
</recv>
<send>
<![CDATA[
SIP/2.0 100 Trying
[last_Via:]
[last_From:]
[last_To:]
[last_Call-ID:]
[last_CSeq:]
Event: presence
Allow: INVITE, ACK, OPTIONS, CANCEL, BYE, SUBSCRIBE, NOTIFY, MESSAGE,
INFO
Content-Length: 0
```

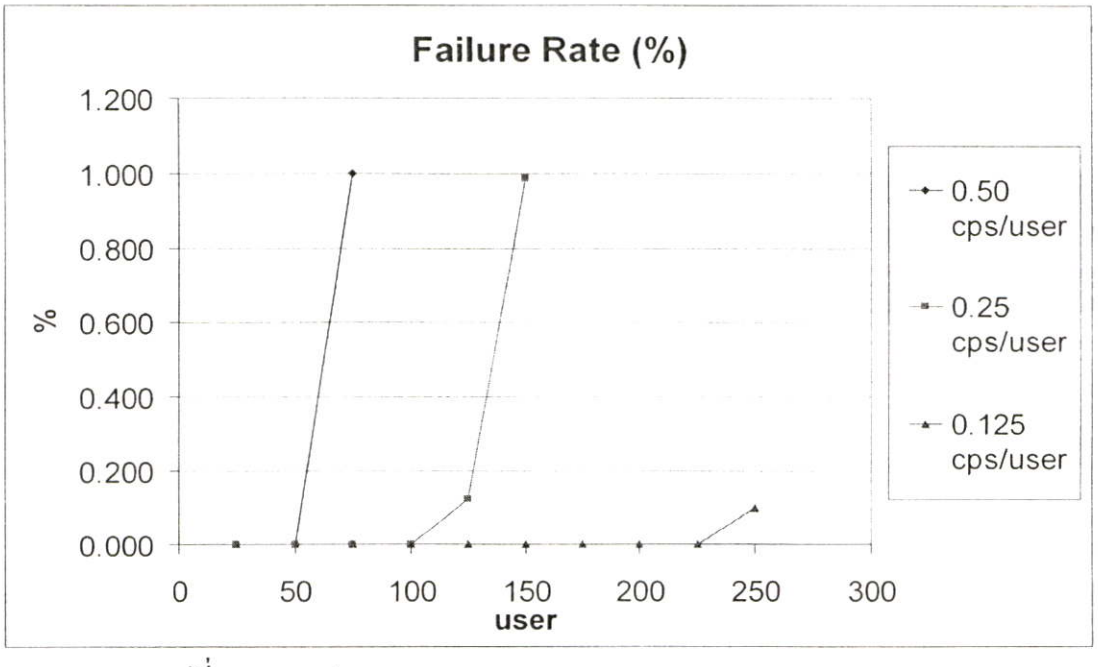
```
]]>
</send>
<send>
<![CDATA[
SIP/2.0 101 Dialog Establishment
[last_Via:]
[last_From:]
[last_To:];tag=[call_number]
[last_Call-ID:]
[last_CSeq:]
Event: presence
Allow: INVITE, ACK, OPTIONS, CANCEL, BYE, SUBSCRIBE, NOTIFY, MESSAGE,
INFO
Content-Length: 0
]]>
</send>
<send>
<![CDATA[
SIP/2.0 202 Accepted subscription
[last_Via:]
[last_From:]
[last_To:];tag=[call_number]
[last_Call-ID:]
[last_CSeq:]
Event: presence
Expires: 3600
Allow: INVITE, ACK, OPTIONS, CANCEL, BYE, SUBSCRIBE, NOTIFY, MESSAGE, INFO
Content-Length: 0
]]>
</send>
<ResponseTimeRepartition value="10, 20, 30, 40, 50, 100, 150, 200"/>
<CallLengthRepartition value="5000,6000,7000,8000,9000,10000"/>
</scenario>
```

เมื่อทำการวัดประสิทธิภาพของ Partysip จะได้ผลลัพธ์รูปของ

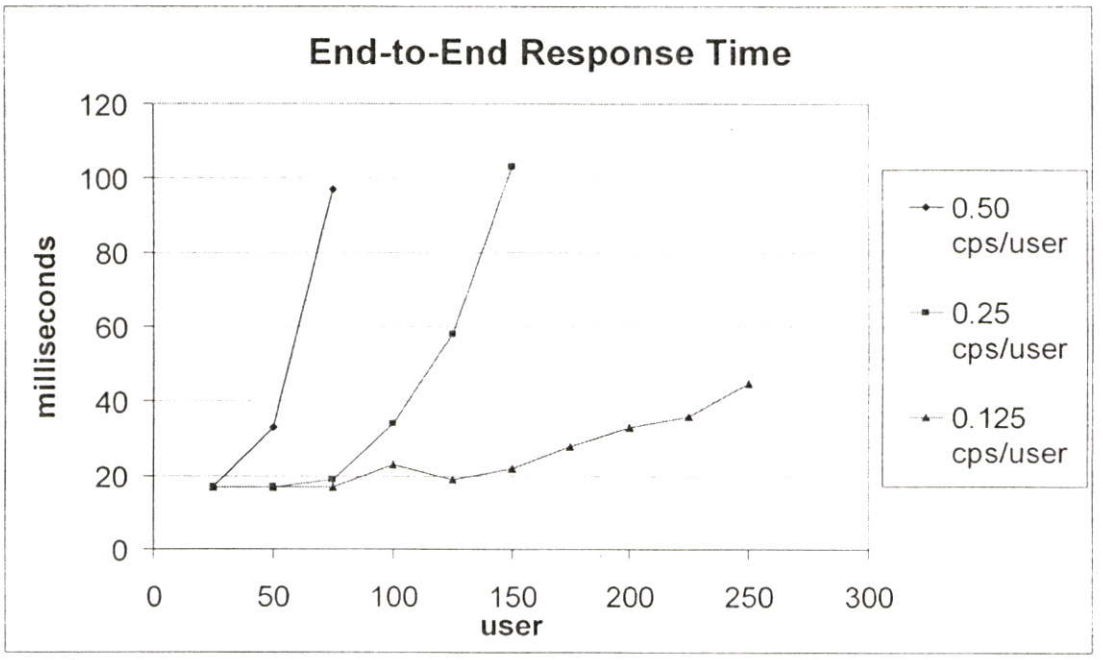
- Failure rate (%) คือ อัตราส่วนของการทำ SUBSCRIBE ไม่สำเร็จ
- End-to-End average response time (ms) คือ ค่าเฉลี่ยของระยะเวลาที่ใช้ไปเมื่อเริ่มทำการส่ง SUBSCRIBE จนกระทั่งได้รับ 101 Dialog Estrablishement กลับมา

ตารางที่ 4.5 ผลลัพธ์การวัดประสิทธิภาพของ Partysip

	0.5 cps/user		0.25 cps/user		0.125 cps/user	
Users	Failed (%)	Response (ms)	Failed (%)	Response (ms)	Failed (%)	Response (ms)
25	0.000	17	0.000	17	0.000	17
50	0.000	33	0.000	17	0.000	17
75	1.000	97	0.000	19	0.000	17
100	-	-	0.000	34	0.000	23
125	-	-	0.125	58	0.000	19
150	-	-	0.987	103	0.000	22
175	-	-	-	-	0.000	28
200	-	-	-	-	0.000	33
225	-	-	-	-	0.000	36
250	-	-	-	-	0.101	45



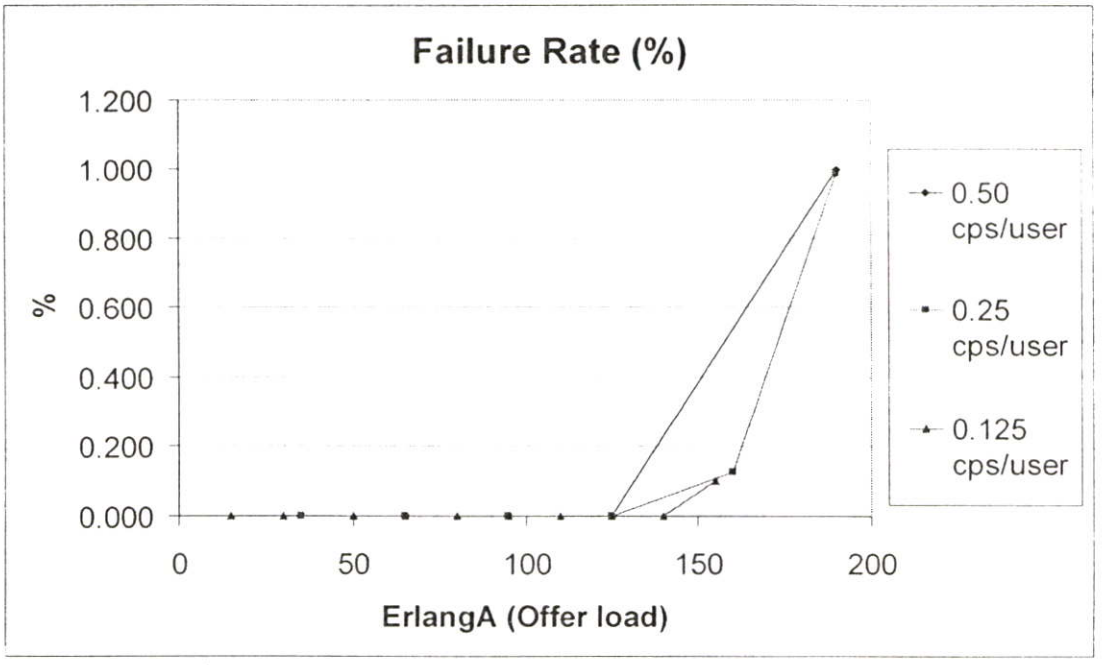
รูปที่ 4.13 กราฟแสดง Failure Rate (%) ของ Partysip (SUBSCRIBE)



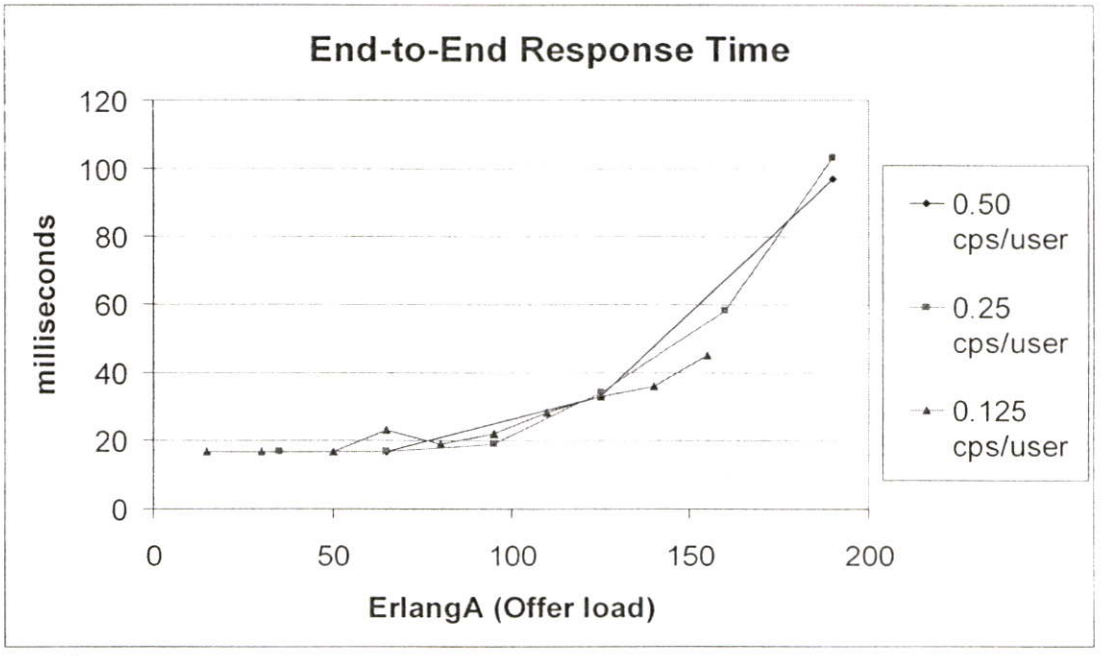
รูปที่ 4.14 กราฟแสดง End-to-End average response time (ms) ของ Partysip (SUBSCRIBE)

ตารางที่ 4.6 ผลลัพธ์การวัดประสิทธิภาพของ Partysip ในลักษณะของ Offer Load

	0.50 cps/user			0.25 cps/user			0.125 cps/user		
User	Erlang A	Fail (%)	ResponseTime (ms)	Erlang A	Fail (%)	ResponseTime (ms)	Erlang A	Fail (%)	ResponseTime (ms)
25	65	0.000	17	35	0.000	17	15	0.000	17
50	125	0.000	33	65	0.000	17	30	0.000	17
75	190	1.000	97	95	0.000	19	50	0.000	17
100	250	-	-	125	0.000	34	65	0.000	23
125	315	-	-	160	0.125	58	80	0.000	19
150	375	-	-	190	0.987	103	95	0.000	22
175	440	-	-	220	-	-	110	0.000	28
200	500	-	-	250	-	-	125	0.000	33
225	565	-	-	285	-	-	140	0.000	36
250	625	-	-	315	-	-	155	0.101	45



รูปที่ 4.15 กราฟแสดง Failure Rate (%) ของ Partysip (SUBSCRIBE)



รูปที่ 4.16 กราฟแสดง End-to-End average response time (ms) ของ Partysip (SUBSCRIBE)

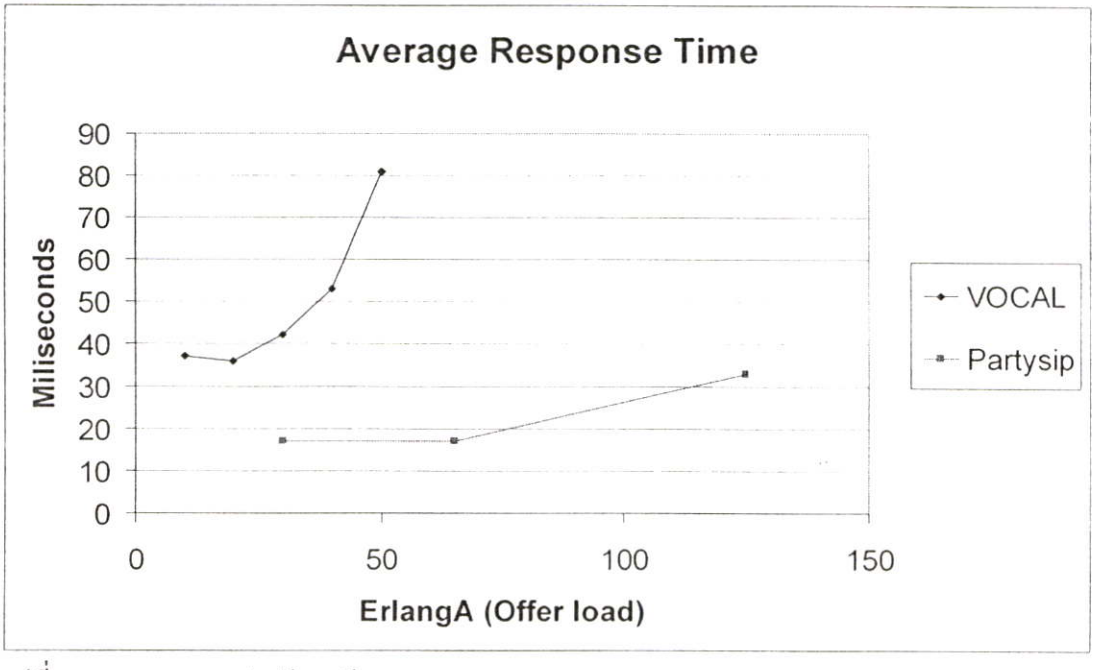
สรุปจากผลการทดลองของ Partysip นั้น จากกราฟจะได้ว่า ไม่ว่าจำนวนผู้ใช้จะเป็นเท่าใดก็ตาม หากมีจำนวน SUBSCRIBE รวมเข้ามากกว่าเกินกว่า 31 cps หรือมี Offer Load มากกว่า 155 Erlang จะทำให้ระบบเริ่มเกิดความผิดหลาดขึ้น ซึ่งหมายความว่า สมมุติหากมีจำนวนผู้ใช้งาน 250 คนก็น่าจะทำการ SUBSCRIBE ไม่เกิน 0.125 cps/user หรือว่าหากมี 125 คนก็ไม่น่าจะเกิน 0.25 cps/user และเซิร์ฟเวอร์จะไม่สามารถทำงานได้เลยหากมีจำนวน SUBSCRIBE รวมเกิน 38 cps หรือมี Offer Load เกิน 190 Erlang ส่วนของ End-to-End average response time

นั้นไม่ขึ้นอยู่กับความถี่การทำ SUBSCRIBE มากกว่าจำนวนผู้ใช้ สำหรับการตั้งค่า retransmission ของ SUBSCRIBE นั้น ยกตัวอย่างเช่นหากมีจำนวนผู้ใช้ 250 คน และทำการส่ง SUBSCRIBE ทุกๆ 8 วินาที (0.125 cps/user) ก็ควรจะตั้งเวลาไว้มากกว่า 45 ms

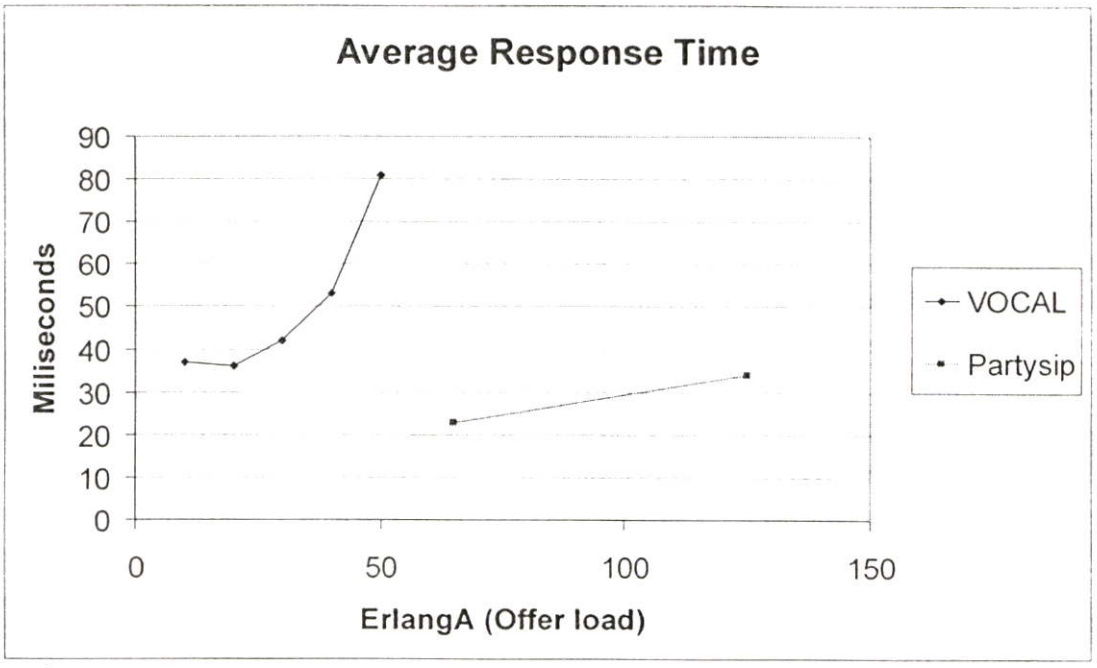
*note : ความแตกต่างของการวัดประสิทธิภาพ INVITE กับ SUBSCRIBE คือ INVITE นั้นผู้ใดในระบบจะทำหรือไม่ทำก็ได้และจะ INVITE ในช่วงเวลาใดก็ได้ ดังนั้นสำหรับ VOCAL จึงต้องมีหน่วยของ INVITE request เป็น call per second (cps) แต่สำหรับ SUBSCRIBE นั้น ผู้ใช้ทุกคนจะต้องทำการ SUBSCRIBE และต้อง SUBSCRIBE ทุกๆช่วงระยะเวลาหนึ่งตามที่ได้กำหนดไว้ ดังนั้นสำหรับ Partysip จึงต้องมีหน่วยของ SUBSCRIBE request เป็น call per second per user (cps/user)

* note : และความแตกต่างที่สำคัญของ VOCAL และ Partysip อีกอย่างคือ VOCAL ไม่รองรับการทำ SUBSCRIBE จึงไม่สามารถวัดผลทางด้านนี้ได้

แต่หากจะนำเอาผลจากทดลองจากทั้งสองตัวอย่างคือ VOCAL และ Partysip ในลักษณะของ Offer Load และ Average Response Time อย่างเดียวแล้วนั้นก็จะได้ผลลัพธ์ดังนี้คือ



รูปที่ 4.17 กราฟเปรียบเทียบ End-to-End Average Response Time ของ VOCAL และ Partysip เมื่อมีจำนวผู้ใช้งานในระบบเป็น 50 users

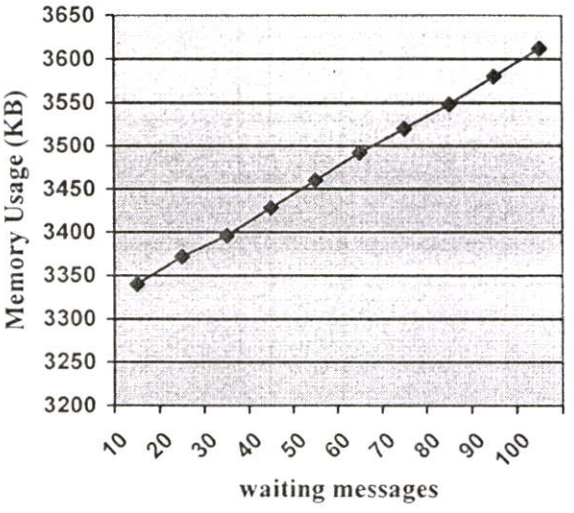


รูปที่ 4.18 กราฟเปรียบเทียบ End-to-End Average Response Time ของ VOCAL และ Partysip เมื่อมีจำนวผู้ใช้งานในระบบเป็น 100 users

ฉะนั้นถ้าจะเปรียบเทียบประสิทธิภาพในด้านเฉพาะของ End-to-End Average Response Time กับจำนวน Offer Load ที่เกิดขึ้นในระบบเพียงอย่างเดียวแล้วนั้น จะเห็นได้ว่า Partysip นั้นมี End-to-End Average Response Time ที่เกิดขึ้นน้อยกว่า VOCAL มาก

4.3 ข้อจำกัดด้านการให้ทรัพยากรของเครื่องของ SIP Traffic Generator and Analyzer

ข้อจำกัดของ CPU ของเครื่องที่ทำการรัน SIP Traffic Generator and Analyzer นั้นไม่มีปัญหา เพราะว่าใช้ CPU ในการทำงานน้อยมากๆและเป็นค่าคงที่คือไม่มีการเปลี่ยนแปลงเลย ส่วนด้านของ memory นั้นขึ้นอยู่กับจำนวนที่ SIP Traffic Generator and Analyzer รอการตอบรับตอบรับกลับมา อย่างเช่น ทำการสร้างสถานการณ์ INVITE request ของ VOCAL ก็จะต้องทำการรอ reponse ที่กลับมาจากอีกฝั่งนึง ซึ่งในที่นี้กว่าจะจบสถานการณ์ที่สร้างขึ้นมาคือ 200 OK ของ BYE ดังนั้นจำนวนที่ INVITE ออกไปและยังไม่ได้รับ 200 OK จึงเป็นตัวที่บอกถึงจำนวน memory ที่ใช้ไป



รูปที่ 4.19 กราฟแสดงจำนวนหน่วยความจำที่ใช้ไปต่อจำนวนของการรอเรสบอนส์

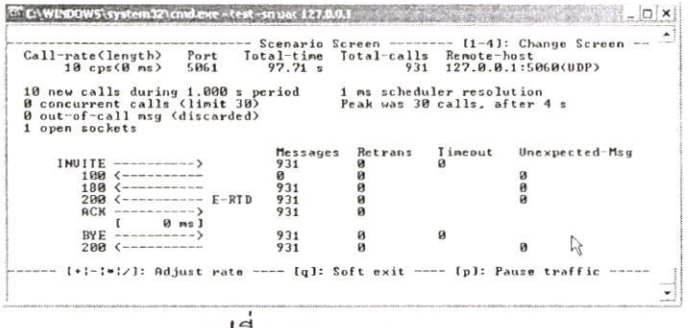
บทที่ 5
สรุป

5.1 สรุปงานวิจัยที่นำเสนอ

วิทยานิพนธ์ฉบับนี้ได้ทำการเสนอ SIP Traffic Generator and Analyzer ซึ่งเป็นเครื่องมือที่ใช้ในการสร้างซิพเมสเสจเพื่อนำมาวัดและวิเคราะห์ประสิทธิภาพของซิพเซิร์ฟเวอร์, อุปกรณ์และเครือข่าย ทั้งนี้ซอฟต์แวร์ได้นำเอา SIPp ซึ่งเป็นโอเพ่นซอร์สมาทำการพอร์ทให้สามารถทำงานได้บนระบบปฏิบัติการที่เป็น Windows และได้เพิ่มประสิทธิภาพของมันโดนการเพิ่ม การสร้างสถานการณ์ให้เป็นมัลติยูซเซอร์ได้ เพิ่มความสามารถให้ส่งซิพเมสเสจผ่าน SIP proxy server หรือ SIP Redirect server ได้ เพิ่มสูตรการคำนวณเพื่อให้ในการวิเคราะห์ประสิทธิภาพคือ Failure rate (%), Offered load (Erlang A), Blocking probability (Erlang B) และ Probability of an arrival call being delayed (Erlang C) โดยที่ยังคงความถูกต้องเดิมไว้ และวิทยานิพนธ์ฉบับนี้ยังได้ยกตัวอย่างการใช้งานโดยนำไปวิเคราะห์ประสิทธิภาพของ VOCAL และ Partysip ด้วย

การวิเคราะห์ประสทธิภาพจากซิพเมสเสจที่ได้ทำการรับและส่งออกจาก SIP Traffic Generator and Analyzer จะแบ่งออกเป็น 4 ด้วยกันคือ

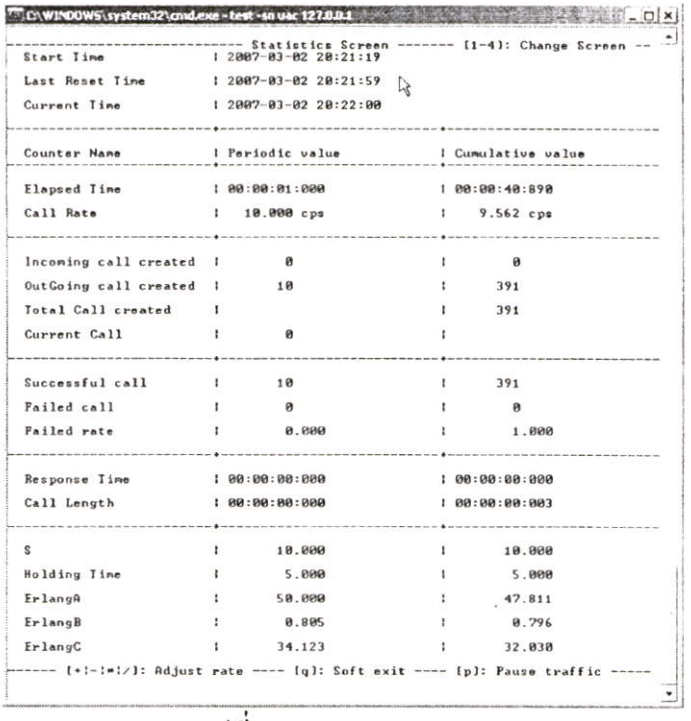
- Scenario screen แสดงจำนวนเมสเสจที่ได้ส่งและรับจากซอฟแวร์ โดยแยกออกเป็นจำนวนที่รับส่งทั้งหมด, จำนวนการทำ retransmission, จำนวนที่เกิด timeout และจำนวนของเมสเสจที่ไม่คาดว่าจะได้รับกลับมา



รูปที่ 5.1 Scenario screen

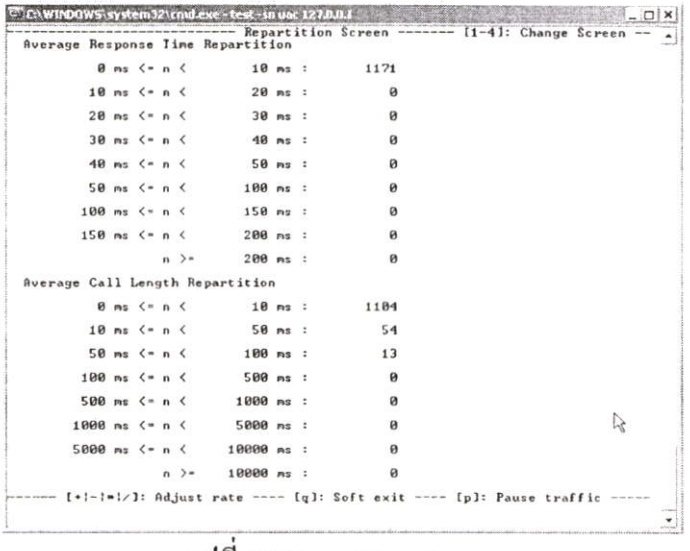
- Statistics Screen แบ่งส่วนแสดงผลออกเป็น 2 ส่วนคือ Periodic value (ค่าที่เกิดขึ้นในช่วงเวลาที่ผ่านมา ซึ่งโดยปกติแล้วจะเป็น 1 วินาทีที่ผ่านมา) และ Cumulative value (คือค่าสะสมสัมพัทธ์ที่เกิดขึ้นตั้งแต่เริ่มทำการรันโปรแกรม) โดยทั้งนี้ทั้ง 2 ส่วนจะแสดง Elapsed time, Call rate, Incoming call created, Outgoing call created, Total call created, Current call, Successful call, Failed call, อัตรา Failed rate (%), ค่าเฉลี่ยของ Response

time และ Call length, Offered load (Erlang A), Blocking probability (Erlang B),Probability of an arrival call being delayed (Erlang C)



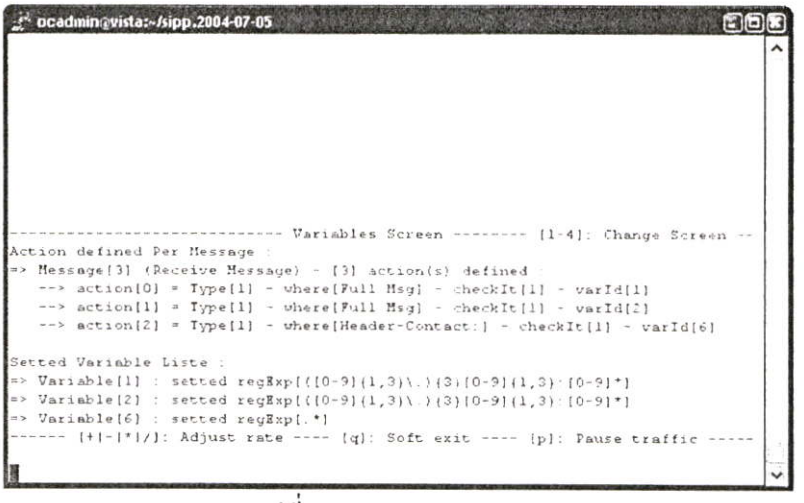
รูปที่ 5.2 Statistics Screen

- Repartition Screen แสดงจำนวนของ Average response time และ Average call length โดยจะทำการแสดงเป็นช่วงเวลาว่าแต่ละช่วงเวลานั้นมีจำนวนเท่าไหร่



รูปที่ 5.3 Repartition Screen

- Variables Screen แสดงข้อมูลของตัวแปรใน scenario ตามที่ระบุไว้ใน regular expression



รูปที่ 5.4 Variables Screen

และสำหรับการสร้างสถานการณ์นั้นจะต้องสร้างเป็นไฟล์ XML โดยการกำหนดฟิลด์ของแต่ละเมสเสจที่จะรับและส่งออกไป ดังนั้นความเหมือนหรือแตกต่างกับสถานการณ์จริงจึงขึ้นอยู่กับความสามารถในการสร้างไฟล์ XML file

ส่วนข้อจำกัดของ CPU ของเครื่องที่ทำการรัน SIP Traffic Generator and Analyzer นั้นไม่มีปัญหา เพราะว่าใช้น้อยมากๆและเป็นค่าคงที่คือไม่มีการเปลี่ยนแปลงเลย ส่วนด้านของ memory นั้นขึ้นอยู่กับจำนวนที่ SIP Traffic Generator and Analyzer รอการตอบรับ ซึ่งมีอัตราความเปลี่ยนแปลงเป็น linear ซึ่งจะเห็นว่าน้อยมากๆเช่นกัน

5.2 ปัญหาและอุปสรรค

เนื่องจากความสามารถของ SIP Traffic Generator and Analyzer ขึ้นอยู่กับการสร้างไฟล์ XML ที่เป็นตัวสร้างสถานการณ์ การใช้งานค่อนข้างจะยาก จำเป็นที่ผู้สร้างสถานการณ์จะต้องทราบถึงแต่ละเฮดเดอร์ฟิลด์และบอดี้ของซิพเมสเสจเป็นอย่างดี และต้องทราบถึงเมสเสจโฟลว์ของสถานการณ์ที่จะสร้างขึ้นด้วย

และการทำงานของระบบซิพที่มีอยู่ในปัจจุบันถึงแม้จะอ้างอิงตามมาตราฐานเดียวกัน แต่ก็มีความแตกต่างในการทำงานอย่างมาก อย่างเช่น VOCAL และ Partysip มีความแตกต่างในด้านเมสเสจโฟลว์และเฮดเดอร์ฟิลด์อยู่พอสมควร จึงไม่สามารถที่จะบอกได้ว่า SIP Traffic Generator and Analyzer นี้นั้นสามารถสร้างสถานการณ์ได้ทุกๆรูปแบบ

5.3 แนวทางการพัฒนาต่อ

ศึกษาถึงระบบของเซิรฟเวอร์ที่มีการใช้งานอยู่เพื่อหาถึงความแตกต่างในด้านการใช้งานแล้วนำมาปรับผรุงให้ซอฟต์แวร์สามารถรองรับการทำงานได้หลากหลายขึ้น เปลี่ยนการแสดงผลจาก command prompt เป็น GUI เพื่อให้สามารถสื่อผลลัพธ์ได้ง่ายและเข้าใจขึ้น เพิ่มให้รองรับการทำแผนเพื่อให้ซอฟต์แวร์สามารถปรับเปลี่ยนสถานการณ์ได้เองตามแผนที่ตั้งไว้

บรรณานุกรม

[1] **RFC3261**, "SIP : Session Initiation Protocol", June 2002

[2] **3GPP TSG SSA, IP Multimedia Subsystem (IMS) – Stage2 (Release 5)**, TS 23.228 v. 5.4.1, 2002-04.

[3] **3GPP TSG CN, Signaling Flows for the IP Multimedia Call Control Based on SIP and SDP - Stage 3 (Release 5)**, TS 24.228 v. 2.0.2, 2002-03.

[4] **3GPP TSG CN, IP Multimedia Call Control Protocol Based on SIP and SDP - Stage 3 (Release 5)**, TS 24.229 v.2.0.1, 2002-03.

[5] Artech House, **SIP Understanding the Session Initiation Protocol**, second edition, 2004

[6] **RFC1889**, "RTP: A Transport Protocol for Real-Time Applications", January 1996

[7] **RFC2326**, "Real Time Streaming Protocol (RTSP)", April 1998

[8] **RFC3015**, "Megaco Protocol Version 1.0", November 2000

[9] **RFC2327**, "Session Description Protocol", April 1998

[10]**RFC3515**, "The Session Initiation Protocol (SIP) Refer Method", April 2003

[11]**RFC3265**, "Session Initiation Protocol (SIP)-Specific Event Notification", June 2002

[12]**RFC3428**, "Session Initiation Protocol (SIP) Extension for Instant Messaging", December 2002

[13]**RFC2976**, "The SIP INFO Method", October 2000

[14]**RFC3262, "**Reliability of Provisional Responses in the Session Initiation Protocol (SIP)**"**, June 2002

[15]**RFC3311**, "The Session Initiation Protocol (SIP) UPDATE Method", September 2002

[16]"**SIPp**", http://sipp.sourceforge.net/

[17]"**Sipsak**", http://sipsak.org/

[18]"**SIPbomber**", http://metalinkltd.com/eng/downloads/

[19]"**SIPSTONE**", http://www.sipstone.org/

[20]"**Winsock**", http://www.sockets.com/

[21]"**PCRE**", PCRE - Perl Compatible Regular Expressions, http://www.pcre.org/

[22]"**UNIXEm : UNIX Emulation Library for Win32**", http://synesis.com.au/software/unixem.html

[23]"**VOCAL**", Vovida Open Communication Application Library, http://www.vovida.org/

[24]"**Partysip**", The partysip SIP proxy server , http://www.nongnu.org/partysip/

ภาคผนวก

ภาคผนวก ก.

วิธีการใช้ SIP Traffic Generator and Analyzer

ก.1 Help (Test -h)

Usage:

test remote_host[:remote_port] [options]

Available options:

-v	: Display version and copyright information.
-bg	: Launch the tool in background mode.
-p local_port	: Set the local port number. Default is a random free port chosen by the system.
-i local_ip	: Set the local IP address for 'Contact:', 'Via:', and 'From:' headers. Default is primary host IP address.
-d duration	: Controls the length (in milliseconds) of of calls. More precisely, this controls the duration of 'pause' instructions in the scenario, if they do not have a 'milliseconds' section. Default value is 0.
-r rate (cps)	: Set the call rate (in calls per seconds). This value can be changed during test by pressing ENTER. Default is 10.
-nu number of user	: Set the number of user (UA#). Default is 10.
-mu max_user	: Set the maximum user (UA#). Default is 10.
-UAS UAS_IP	: Set the UAS IP address for 'Route:' header Default is 127.0.0.1.
-hold holding time	: Set the holding time for calculating Erlang Default is 5 sec
-sf filename	: Loads an alternate xml scenario file.

To learn more about XML scenario syntax,
use the -sd option to dump embedded
scenarios. They contain all the necessary
help.

-sn name : Use a default scenario (embedded in
the SIP_CGA executable). Available values
in this version:

'uac' : Standard SipStone UAC (default).

'uas' : Simple UAS responder (UDP only).

'regexp' : Standard SipStone UAC - with
regexp and variables.

-sd name : Dumps a default scenario (embeded in
the SIP_CGA executable)

-t [u1|un|t1|tn] : Set the transport mode:

u1: UDP with one socket (default),

un: UDP with one socket per call,

t1: TCP with one socket,

tn: TCP with one socket per call.

It appears that you installed the
sippcomp.so pluggin. 2 additionnal
transport modes are available:

c1: u1 + compression,

cn: un + compression.

-trace_msg : Displays sent and received SIP messages in
sipp_messages.log

-trace_stat : Dumps all statistics in the <scenario_name.csv>
file. Use the '-h stat' option for a detailed
description of the statistics file content.

-stf file_name : Set the file name to use to dump statistics

-trace_err : Trace all unexpected messages in
sipp_errors.log.

-s service_name : Set the username part of the resquest URI.
Default is 'service'.

-f frequency	: Set the statistics report frequency on screen (in seconds). Default is 1.
-fd frequency	: Set the statistics dump log report frequency (in seconds). Default is 60.
-l calls_limit	: Set the maximum number of simultaneous calls. Once this limit is reached, traffic is decreased until the number of open calls goes down. Default: (3 * call_duration (s) * rate).
-m calls	: Stop the test and exit when 'calls' calls are processed.
-mp local_port	: Set the local RTP echo port number. Default is none. RTP/UDP packets received on that port are echoed to their sender.
-mi local_rtp_ip	: Set the local IP address for RTP echo.
-nr	: Disable retransmission in UDP mode.

Exit code:

Upon exit (on fatal error or when the number of asked calls (-m option) is reached, SIP_CGA exits with one of the following exit code:

0: All calls were successful

1: At least one call failed

99: Normal exit without calls processed

-1: Fatal error

Example:

Run with embedded server (uas) scenario:

test -sn uas

On the same host, run SIP_CGA with embedded client (uac) scenario

test -sn uac 127.0.0.1

ก.2 วิธีการเขียน XML file เพื่อใช้สร้างสถานการณ์

Scenario ของ SIP_CGA จะถูกเขียนในรูปแบบที่เป็น XML และต้องขึ้นต้นด้วยรูปแบบดังนี้เสมอ

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<scenario name="Basic Sipstone UAC">
```

และจบด้วย

```
</scenario>
```

ตารางที่ ก.1 ลิสต์คำสั่งของและแอททริบิวต์

Command	Attribute	Description	Example
<send>	retrans	ถูกใช้ใน UDP เท่านั้น กำหนดระยะห่างเริ่มต้นในการ retransmission โดยที่จะทำการ retransmission ทั้งหมด 7 ครั้ง ก่อนจะยกเลิกการ call นั้นๆ หน่วยเป็น ms (Exponential Backoff Interval)	<send retrans="500"> Retranmission หลังเวลา 500ms, 1000ms, 2000ms, 4000ms, 8000ms, 16000ms และ 32000ms
	start_rtd	ระบุจุดเริ่มต้นจับเวลาหลังจาก message ถูกส่ง โดยปกติจะเริ่มที่ message แรก ของ scenario	<send start_rtd="true"> เริ่มต้นจับเวลาหลังจาก message นี้ถูกส่ง
	rtd	หยุดจับเวลา	<send rtd="true"> หยุดเวลาหลังจาก message นี้ถูกส่ง
	lost	จำลองอัตรา packet lost	<send lost="10"> 10% ของ message นี้จะไม่ถูกส่ง
<recv>	response	ระบุว่าต้องการ SIP message ที่มี code อะไร	<recv response="200"> SIP_CGA ต้องการรับ SIP message ที่มี code = “200”
	request	ระบุว่าต้องการ SIP message request อะไร	<recv response="ACK"> SIP_CGA ต้องการ “ACK” SIP message

Command	Attribute	Description	Example
	optional	ระบุว่า message ที่ต้องการรับนั้นเป็นเพียง option คือไม่จำเป็นว่าต้องได้รับ	<recv response="100" optional="true">
	rrs	Record Route Set เป็นการเก็บค่าเฮเดอร์ “Record-Route:” ของ message ที่ได้รับ และสามารถเรียกใช้ได้โดย [Routes]	<recv response="100" rrs="true">
	start_rtd	ระบุจุดเริ่มต้นจับเวลาหลังจากได้รับ message โดยปกติจะเริ่มที่ message แรก ของ scenario	<recv start_rtd="true"> เริ่มจับเวลาเมื่อได้รับ message นี้
	rtd	หยุดจับเวลา	<recv rtd="true"> หยุดเวลาหลังจากได้รับ message นี้
	lost	จำลองอัตรา packet lost	<recv lost="10"> 10% ของ message นี้ที่ได้รับจะถูกทิ้งไป
	action	ระบุ action ถ้าได้รับ message นี้	ตัวอย่าง regular expression action <recv response="200"> <action> <ereg regexp="([0-9]{1,3}\.){3}[0-9]{1,3}:[0-9]*" search_in="msg" check_it="true" assign_to="1,2"/> </action> </recv>

Command	Attribute	Description	Example
<sendCmd>	<![CDATA[ ]]>	เนื้อหาที่จะถูกส่งไปที่ 3PCC SIP_CGA และประกอบด้วย Call-ID	<sendCmd> <![CDATA[Call-ID: [call_id] [$1]]]> </sendCmd>
<recvCmd>	action	ระบุ action เมื่อได้รับ command	ตัวอย่าง regular expression ที่จะได้รับจาก SendCmd <recvCmd> <action <ereg regexp="Content-Type:.*" search_in="msg" assign_to="2"/> </action> </recvCmd>
<pause>	milliseconds	ระบุ delay ในหน่วย ms ถ้าค่านี้ไม่ระบุจะใช้ค่าพารามิเตอร์ใน –d command line	<pause milliseconds="5000"> หลุดหรือหน่วงเวลา scenario นี้เป็นเวลา 5 วินาที
<ResponseTimeRepartition>	value	ช่วงเวลาในหน่วย ms ซึ่งถูกใช้เพื่อ distribute เวลาในการ reponse	<ResponseTimeRepartition value="10, 20, 30"/> เวลาของการ response จะถูก distribute ในช่วง 0-10ms, 10-20ms, 20-30ms และมากกว่า 30ms

Command	Attribute	Description	Example
<CallLengthRepartition>	value	ช่วงเวลาในหน่วย ms ซึ่งถูกใช้เพื่อ distribute ระยะเวลาในการ Call	<CallLengthRepartition value="10, 20, 30"/> เวลาของการ response จะถูก distribute ในช่วง 0-10ms, 10-20ms, 20-30ms และมากกว่า 30ms จะถูก distribute ในช่วง 0-10ms, 10-20ms, 20-30ms และมากกว่า 30ms

ตารางที่ ก.2 ลิสต์ของ Key word ภายใน <SendCmd> ระหว่าง <![CDATA[และ]]>

Keyword	default	Description
[service]	service	ถูกส่งมาจาก –s service name
[remote_ip]	-	Remote IP address ถูกส่งมาจาก command line
[remote_port]	5060	Remote IP address ถูกส่งมาจาก command line
[transport]	UDP	ขึ้นอยู่กับค่าพาพรามิเตอร์จาก –t คือ TCP หรือ UDP
[local_ip]	Primary host IP address	ได้ค่าจากพารามิเตอร์ -i
[local_port]	Random	ได้ค่าจากพารามิเตอร์ -p
[call_number]	-	เป็น index มีค่าเริ่มต้นเป็น 1 และจะเพิ่มขึ้นทีละ 1 ต่อการ call แต่ละครั้ง
[call_id]	-	เป็น identifier ของแต่ละการ call ซึ่งจะถูก generate โดย SIP_CGA เอง ซึ่งจะถูกใช้ในเฮดเดอร์ “Call-ID”
[media_ip]	-	ขึ้นอยู่กับค่าพารามิเตอร์จาก –mi ซึ่งเป็น local IP address สำหรับ RTP echo
[media_port]	-	ขึ้นอยู่กับค่าพารามิเตอร์จาก –mp เป็น local RTP echo port ไม่มีค่า default RTP/UDP packet จะได้รับจากพอร์ทนี้และ echo กลับไปยังผู้ส่ง

Keyword	default	Description
[last_*]	-	[last_*] จะถูกแทนที่อัตโนมัติด้วยค่าจากเฮดเดอร์ของ message ที่ได้รับเป็น ทำ message สุดท้าย ยกเว้น message จากการ retransmission ถ้าเฮดเดอร์ในส่วนนั้นไม่ได้ระบุไว้แล้วค่าของ [last_*] จะถูกเพิกเฉยไว้ และถ้าเฮดเดอร์นั้นถูกระบุอยู่หลายๆที่ ค่าของมันจะถูก [last_*] นำมาต่อๆกัน ซึ่งจะต้องใช้ [CRLF] เป็นตัวแยก
[field0-n]	-	ใช้เพื่อดึงค่ามาจาก CSV file ภายนอก
[$n]	-	ใช้เพื่อดึงค่าหมายเลขของการ Call n
[user_id]	-	เป็นหมายเลขของ user ซึ่งจะเพิ่มขึ้นทีละหนึ่งตามการ call แต่ละครั้ง โดยมีค่าสูงสุดอยู่ที่ค่าพารามิเตอร์ของ –mu เมื่อถึงค่าสูงสุดจะย้อนกลับมาเริ่มต้นที่ 1 ใหม่
[UAS_IP]	127.0.0.1	IP address ของ SIP_CGA อีกตัวหนึ่งที่ทำงานเป็นคู่สื่อสาร

Regular Expression

การใช้ regular expression ทำให้ SIP_CGA สามารถ

- ขยายเนื้อหาภายใน message หรือ header ของ SIP และเก็บมันไว้เพื่อการใช้งานในภายหน้า
- ตรวจสอบว่าส่วนหนึ่งภายใน message หรือ header ของ SIP นั้นตรงกับ expression ที่ต้องการ

Regular expression ที่ถูกใช้ใน SIP_CGA ถูกนิยามไว้ใน Posix Extended standard (POSIX 1003.2)

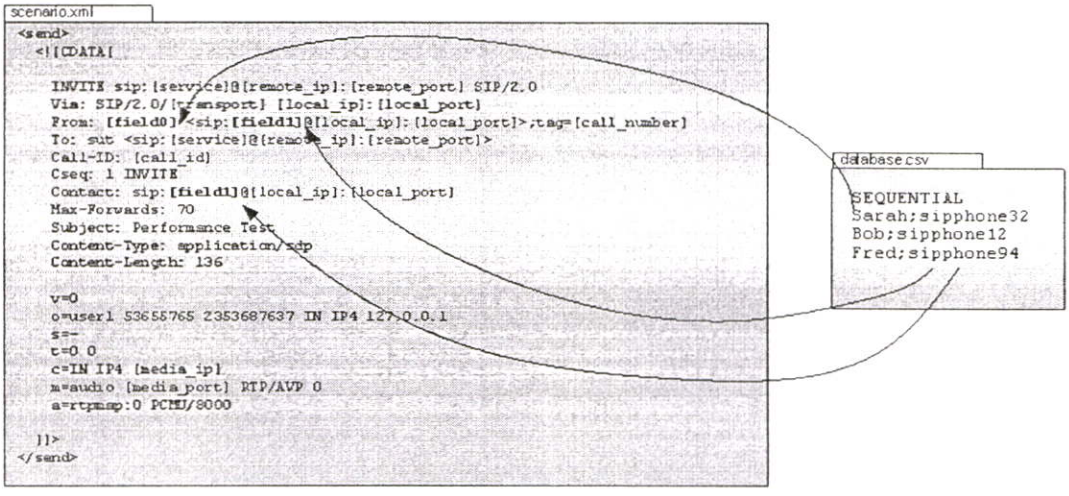
ตารางที่ ก.3 Regular Ezpression

Keyword	Default	description
regexp	None	บรรจุ regular expression ที่จะใช้สำหรับ matching กับ message หรือ header ที่ได้รับ
search_in	msg	มี 2 ค่า คือ “msg” เพื่อ match กับทั้ง message และ “hdr” เพื่อ match กับเฉพาะส่วนของ header ที่กำหนดเท่านั้น
header	None	จะใช้ก็ต่อเมื่อ search_in มีค่าเป็น “hdr” เพื่อใช้ระบุส่วนของ header ที่ต้องการ
check_it	false	ถ้ามีค่าเป็น true การ call นี้จะถูก mark ถ้า regexp ไม่ match

Keyword	Default	description
assign_to	None	บรรจุตัวแปร id(integer) หรือลิสต์ของตัวแปร id เพื่อใช้เก็บค่าจากการ matching ซึ่งค่านี้สามารถนำออกมาใช้ภายหลังได้ด้วย [$n] โดยที่ n คือ id

การดึงค่าจาก CSV ไฟล์ภายนอกระหว่างการ call

สามารถใช้ –inf file_name ที่ command line เพื่อทำการอินพุทค่าใส่ใน scenario โดยที่ในบรรทัดแรกจะต้องระบุว่าข้อมูลที่จะอ่านนั้นจะเป็นแบบมีลำดับ SEQUENTIAL หรือสุ่ม RANDOM แต่ละบรรทัดสัมพันธ์กับ 1 call เท่านั้น ถ้ามีมากกว่า 1 ให้ใช้ ; เป็นตัวคั่น และใน scenario ไฟล์นั้นจะใช้ [field0], [field1], ... เป็นตัวอ้างถึง



รูปที่ ก.1 การดึงค่าจาก CSV ไฟล์ภายนอกระหว่างการ call

ก.3 Traffic Control

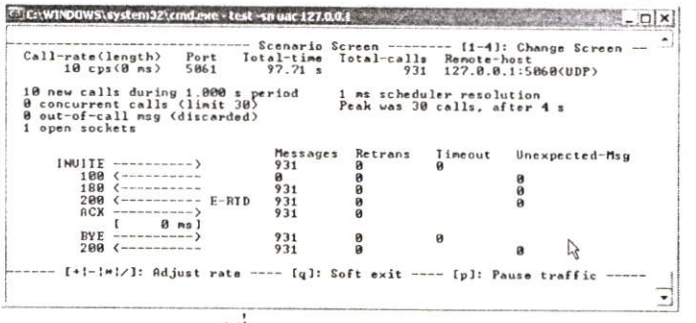
สามารถควบคุมจำนวนการ call ในระหว่างการใช้งานได้ด้วย key ดังต่อไปนี้

- \+ เพิ่มอัตราการ call อีก 1
- \- ลดอัตราการ call อีก 1
- * เพิ่มอัตราการ call อีก 10
- / ลดอัตราการ call อีก 1
- P หยุดการ call ชั่วคราว และรอจนกระทั่งการ call ปัจจุบันนั้นเสร็จสิ้นลง
- Q ยกเลิกการ call แต่ยังรอจนกระทั่งการ call ปัจจุบันนั้นเสร็จสิ้นลง

ก.4 Screen

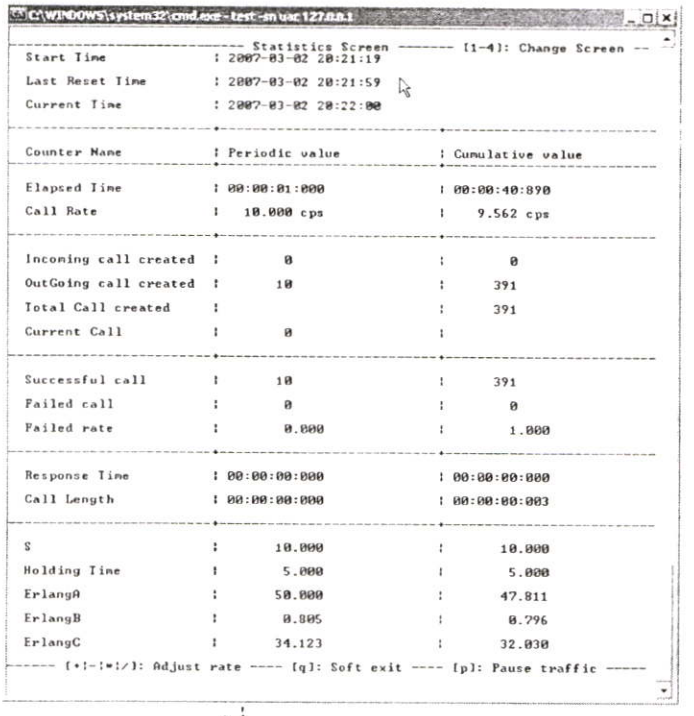
สามารถเปลี่ยนรูปแบบการแสดงผลได้ด้วยการกด key '1', '2', '3' และ '4'

- Key '1' Scenario Screen แสดง call flow ของ scenario นั้นและข้อมูลที่สำคัญ



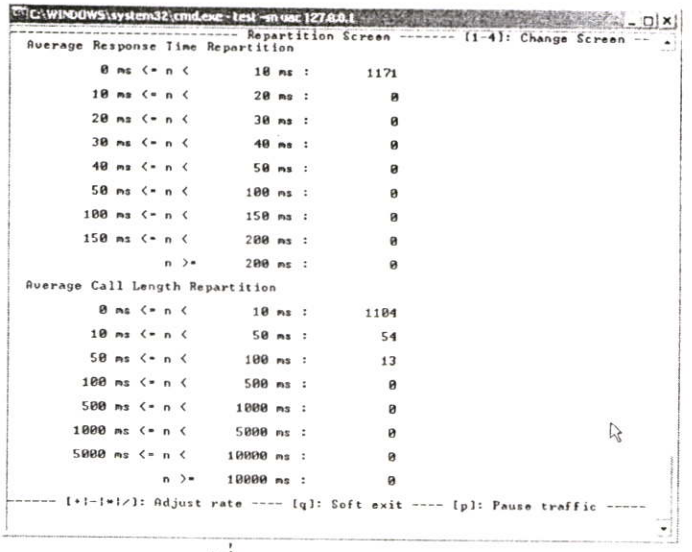
รูปที่ ก.2 Scenario Screen

- Key '2' Statistics Screen แสดงสถิติหลักโดยแบ่งเป็นสองส่วน ส่วนแรกคือสถิติตั้งแต่เริ่มการ call แรก ส่วนหลังคือสถิติเฉพาะในช่วงเวลาที่กำหนด โดยสามารถกำหนดช่วงเวลาได้ด้วยค่าพารามิเตอร์ –f จาก command line



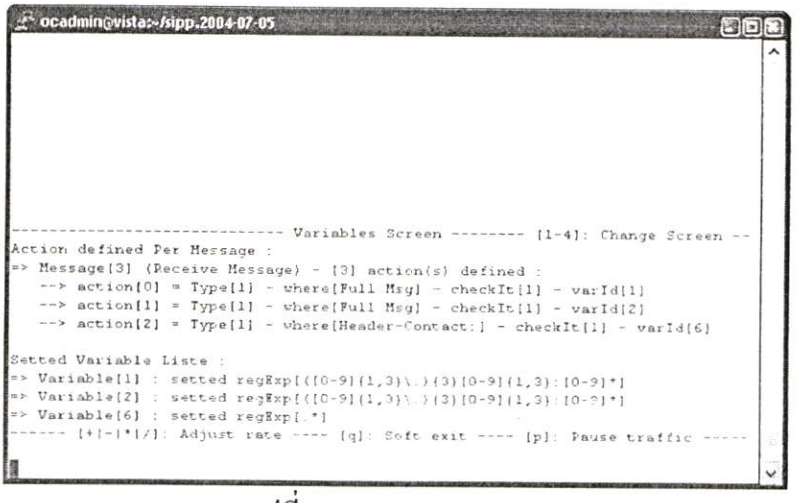
รูปที่ ก.3 Statistics Screen

- Key '3' Repartition Screen แสดงการกระจายตัวของ response timeและ call length ตามที่ระบุไว้ใน scenario



รูปที่ ก.4 Repartition Screen

- Key '4' Variables Screen แสดงข้อมูลของตัวแปรใน scenario ตามที่ระบุไว้ใน regular expression



รูปที่ ก.5 Variables Screen

ภาคผนวก ข.

ผลงานวิจัยที่ได้รับการตีพิมพ์เผยแพร่

1. วุฒินัย กาญจนศร, "ซิพซีจีเอ : ซิพคอลล์เจเนอเรเตอร์และอนาไลเซอร์", Electrical Engineering Conference 28, EECON28 , 20-21 October 2005, Thailand

2. Wutthinai Kanjanasorn and Surin Kittitornkun, "SIP_CGA : SIP Call Generator and Analyzer", The 9th National Computer Science and Engineering Conference University of the Thai Chamber of Commerce, NCSEC2005, 27-28 October 2005, Bangkok, THAILAND

การประชุมวิชาการทางวิศวกรรมไฟฟ้า ครั้งที่ 28

28th Electrical Engineering Conference (EECON 28)

ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยธรรมศาสตร์

สารบัญบทความ

สารบัญ

สาขาบทความ CP

วิศวกรรมคอมพิวเตอร์และเทคโนโลยีสารสนเทศ (Computer Engineering and Information Technology)

ซิพซีจีเอ : ซิพคอลล์เจเนอเรเตอร์และอนาไลเซอร์

SIP_CGA : SIP Call Generator and Analyzer

วุฒินัย กาญจนศร และ สุรินทร์ กิตติธรกุล
ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
3 หมู่ 2 ถนนฉลองกรุง เขตลาดกระบัง กรุงเทพ 10520
โทร. 0-2739-2400-1 โทรสาร 0-2739-2404 E-mail:wutthinai@yahoo.com, kksurin@kmitl.ac.th

CP007

บทคัดย่อ

ซิพคอลล์เจเนอเรเตอร์และอนาไลเซอร์เป็นโปรแกรมที่ใช้งานสำหรับทดสอบประสิทธิภาพของ SIP server และอุปกรณ์ต่างๆที่เกี่ยวข้อง ซึ่งโปรแกรมจะจำลองสถานการณ์ให้เหมือนกับกำลังมีการใช้งานจำนวนมากๆได้ โดยที่มีความใกล้เคียงกับการใช้งานจริงหรือสามารถสร้างสถานการณ์ขึ้นเองได้ตามความต้องการ และมีการวิเคราะห์หรือแสดงผลที่เกิดขึ้นจากการทดลองได้ ทั้งนี้ซิพคอลล์เจเนอเรเตอร์และอนาไลเซอร์ได้ทำการพัฒนาต่อมาจาก SIPp ที่เป็น open source แล้วทำการพอร์ทจากที่ทำงานบนลีนุกซ์มาเป็นทำงานบนวินโดวส์ ซึ่งยังได้เพิ่มความสามารถให้ทำงานได้ใกล้เคียงกับมาตรฐานของซิพมากขึ้นและให้สามารถจำลองสถานการณ์เป็นผู้ใช้หลายๆคนได้ ซึ่งโปรแกรมสำหรับทดสอบอื่นๆนั้นยังไม่สามารถทำได้ โดยที่เอกสารฉบับนี้จะอธิบายถึงลักษณะการใช้งานของ SIP_CGA คุณสมบัติของ SIP_CGA โดยเปรียบเทียบกับโปรแกรมอื่นๆ การพัฒนาและลักษณะการทำงานของส่วนต่างๆในโปรแกรม ผลการทดลองนำไปใช้งานจริง และส่วนสุดท้ายคือสรุปผลจากการทดลองนำใช้งาน

คำสำคัญ : SIP, คอลล์เจเนอเรเตอร์, อนาไลเซอร์, ประสิทธิภาพ, SIPp

Abstract

SIP Call Generator and Analyzer is used for testing SIP server and analysing its performance. It can emulate almost every real situations. SIP tester SIP_CGA is developed on SIPp which is an open source program which running on Linux system and ported to run on Windows XP. SIP_CGA has been improved to comply with SIP standard and emulate multiuser situations that other programs can not. This paper explains how to use SIP_CGA, its specification by comparing with other similar programs, its structure and development. Experimantal results are shown and concluded

Keywords : SIP, Call Generator, Analyzer, performance, SIPp

1. บทนำ

เนื่องจากการใช้งาน VoIP (Voice Over IP) กำลังเป็นที่นิยมแพร่หลายมากขึ้นในปัจจุบัน และ SIP (Session Initiation Protocol) RFC3261 [1] ก็เป็น IETF โปรโตคอลที่ทำงานอยู่บนชั้นแอพพลิเคชั่นเลเยอร์ซึ่งใช้สำหรับควบคุมการสร้าง, แก้ไขและทำลายเซสชั่นของการติดต่อสื่อสารของคู่สื่อสารนั้น ซึ่ง SIP นั้นทำให้ขอบเขตของการใช้งานของแอพพลิเคชั่นไม่ได้จำกัดอยู่เฉพาะสัญญาณเสียงเท่านั้น แต่ยังขยายขอบเขตให้สามารถใช้งานรับส่งมัลติมีเดียผ่านไอพีได้ (กลายเป็น Instant Messaging หรือ Presence) และในปี 2000 นั้น 3GPP (Third Generation Partnership Program) ได้เลือกใช้ SIP เพื่อเป็นโปรโตคอลเซสชั่นสำหรับเครือข่ายโมบายล์ไอพีในยุคสามจี (3GPP Release 5) [2], [3], [4]

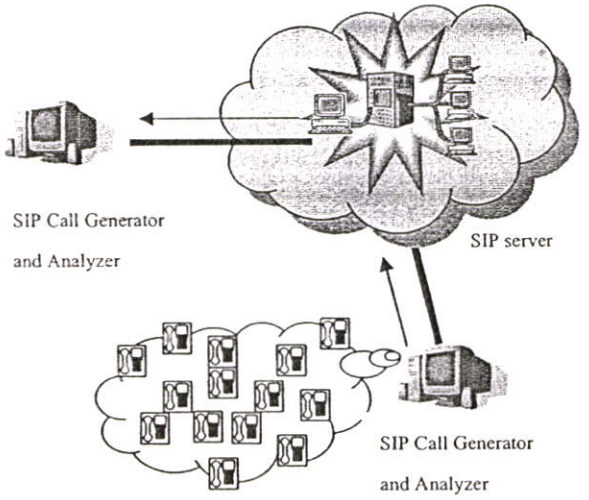
ดังนั้นเมื่อ SIP เริ่มมีการใช้งานกันมากขึ้น จึงจำเป็นที่จะต้องมีเครื่องมือที่ใช้สำหรับทดสอบความสามารถหรือประสิทธิภาพในการทำงานของ SIP server และเครือข่ายที่ใช้งานนั้นๆได้ เพื่อนำไปปรับปรุงการตั้งค่าพารามิเตอร์ต่างๆให้สามารถรองรับจำนวนผู้ใช้งานได้มากขึ้น และมีประสิทธิภาพในการใช้งานของแต่ละผู้ใช้มากขึ้น และเพื่อวิเคราะห์ถึงปัญหาที่อาจจะเกิดขึ้นในส่วนต่างๆได้ เช่น ปัญหาคอขวดทางด้านประสิทธิภาพของเครือข่ายที่จุดต่างๆ ซึ่ง SIP Call Generator and Analyzer ที่มีอยู่ในปัจจุบันนั้นมักไม่เป็นไปตามมาตรฐาน RFC3261 ส่วนใหญ่ไม่สามารถจำลองการใช้งานเป็นหลายๆผู้ใช้งานในขณะเดียวกันได้ และไม่สามารถที่จะทำการสร้างแพ็กเก็ตเพื่อใช้งานผ่านทางหรือกซี่จากผู้ใช้คนหนึ่งไปยังอีกคนหนึ่งได้ รวมทั้งเครื่องมือนี้ส่วนใหญ่จะอยู่ในระบบปฏิบัติการที่เป็นยูนิกซ์อีกด้วย จึงควรที่จะมี SIP Call Generator and Analyzer ที่นำมาใช้บนระบบปฏิบัติการที่เป็นวินโดวส์บ้าง และมีความสามารถที่จะสร้างแพ็กเก็ตหรือสถานการณ์ขึ้นมาใหม่เองให้เป็นไปตามมาตรฐาน และมีความเหมือนหรือคล้ายคลึงกับการใช้งานจริงๆให้มากที่สุด

2. งานวิจัยที่เกี่ยวข้องและคุณสมบัติของ SIP_CGA

2.1 หลักการทำงานทั่วๆไปของ SIP Call Generator and Analyzer

SIP Call Generator and Analyzer เป็นเครื่องมือที่ใช้ในการสร้างแพ็กเก็ตจำนวนมากของ SIP เสมือนเป็นการจำลองว่าเป็นการใช้

งานจาก SIP phone จำนวนมาก เพื่อเป็นการทดสอบประสิทธิภาพของเซิร์ฟเวอร์หรือของเครือข่าย และทำการวัดถึงผลลัพธ์ที่เกิดขึ้นจากสถานการณ์ที่สร้างขึ้นจาก SIP Call Generator and Analyzer ที่ใช้ในการทดสอบนั้นๆ เพื่อนำไปปรับปรุงอุปกรณ์ต่างๆ หรือเพื่อปรับแก้ถึงค่าพารามิเตอร์ที่ใช้ เช่น อายุของการทำ REGISTER แต่ละครั้งว่าควรจะเป็นเท่าไหร่ ผู้ใช้ควรจะทำการ REGISTER ทุกๆกี่วินาทีเพื่อให้เหมาะสมกับอายุของมัน และเพื่อวัดถึงดีเลย์และ error หรืออัตราความล้มเหลวของแพ็กเก็ตที่เกิดขึ้นเพื่อกำหนดอายุที่เหมาะสมของการทำ REGISTER และกำหนดจำนวนของการ retransmission ของ REGISTER แพ็กเก็ตเมื่อกระทำการร้องขอไม่สำเร็จ



รูปที่ 1 การทำงานของ SIP Call Generator and Analyzer

จากรูปคือ SIP Call Generator ทั้ง 2 ตัว จะทำการโต้ตอบกันตามสถานการณ์ที่ได้กำหนดไว้ โดยทำการสื่อสารกันผ่านเครือข่ายต่างๆ โดยที่มี SIP server เป็นผู้ประสานงานการสื่อสารหรือเป็นผู้ควบคุม

2.2 การเปรียบเทียบคุณสมบัติของโปรแกรม

ทำการเปรียบเทียบคุณสมบัติความสามารถระหว่าง SIP_CGA กับโปรแกรมอื่นๆที่มีอยู่ เพื่อให้ทราบถึงคุณสมบัติเด่นของแต่ละโปรแกรม โดยโปรแกรมที่นำมาเปรียบเทียบมีดังนี้

1. SIPp 1.0 (SP) [5]
2. Sipbomber 0.7 (SB) [6]
3. Sipsak 0.8.12 (SS) [7]

ตารางที่ 1 เปรียบเทียบคุณสมบัติความสามารถของแต่ละโปรแกรม

คุณสมบัติและความสามารถ	SP	SB	SS	Ours
ภาษาที่พัฒนา	C/C++	Java	C++	C/C++
Platform	Linux	Redhat9	Linux	Windows
สร้างแพ็กเก็ตจำนวนมากได้	✓	✓	✓	✓
ทดสอบระหว่าง UAS และ UAC	✓		✓	✓
สร้างรูปแบบสถานการณ์เองได้	✓	✓		✓
ส่งแพ็กเก็ตผ่าน SIP server ต่างๆ	บางแพ็กเก็ต	บางแพ็กเก็ต	บางแพ็กเก็ต	บางแพ็กเก็ต
สร้างสถานการณ์เป็น Multiuser				✓
โปรโตคอลในชั้น Transport	TCP/UDP	TCP/UDP	UDP	TCP/UDP
GUI		✓		
คำนวณสถิติ	✓	✓		✓

ทั้งนี้การส่งแพ็กเก็ตผ่าน SIP server ซึ่งได้เฉพาะบางแพ็กเก็ตนั้นเนื่องจาก ส่วนของฟิลด์บางฟิลด์ในเฮดเดอร์ของแต่ละโปรแกรมยังมีปัญหาอยู่เช่นมีฟิลด์ไม่ครบหรือไม่เป็นมาตรฐาน ทำให้ความสามารถในการส่งแพ็กเก็จจากผู้ใช้คนหนึ่งไปยังอีกคนหนึ่งโดยผ่านทางเซิร์ฟเวอร์นั้นยังมีปัญหาอยู่ คือ Sipsak นั้นไม่มีฟิลด์ "Route:" ในส่วนของเฮดเดอร์จึงไม่สามารถส่งแพ็กเก็ตระหว่างไคลเอนต์ผ่านทางเซิร์ฟเวอร์ได้ และสำหรับ SIPbomber ถูกทำมาเพื่อทดสอบการอิมพลีเมนต์ของเซิร์ฟเวอร์เท่านั้นจึงไม่สามารถจำลองการสื่อสารระหว่างไคลเอนต์ได้

3. การพัฒนาโปรแกรมและลักษณะการทำงาน

สำหรับการพัฒนานั้นได้นำโปรแกรมชื่อ SIPp เวอร์ชั่น 1.0 ซึ่งเป็นโปรแกรม open source ที่เขียนด้วยภาษา C/C++ มา ซึ่งทำงานอยู่บนระบบปฏิบัติการลีนุกซ์ตระกูล Redhat โดยการทำงานจะอยู่บน Terminal ซึ่งเราได้นำมาพอร์ทและแก้ไขเพิ่มเติมลงบน Microsoft Windows XP โดยใช้ Microsoft Visual C++ version 6 service pack 5 เป็นเครื่องมือ และให้มันสามารถทำงานบน Command Prompt บนวินโดวส์ได้

และทั้งนี้ SIPp นั้นเป็นโปรแกรม open source ที่นำเอาความสามารถพื้นฐานมาจากโปรแกรมชื่อ SIPSTONE [8] ที่พัฒนาโดย

Columbia University และ Ubiquity ซึ่งเป็นผู้ริเริ่มและผลักดันให้โปรโตคอล SIP เกิดขึ้นมาและถูกใช้เป็นมาตรฐานขึ้น

โดยที่พวกเราได้ทำการเพิ่มความสามารถให้สามารถจำลองสถานการณ์เป็น Multiuser ให้มีความเหมือนกับการใช้งานจริง คือ เหมือนกับมีหลายผู้ใช้ต่างๆกันเข้ามาใช้งานจำนวนมากแทนที่จะเป็นผู้ใช้เดียวกัน เนื่องจากหากเป็นการทำงานของผู้ใช้คนเดียวนั้น SIP server อาจจะทำการประมวลผลการร้องขอหรือตอบกลับเพียงครั้งเดียวเพราะเห็นว่าส่งมาจากหรือส่งไปยังผู้ใช้เดียวกัน ทั้งนี้ขึ้นอยู่กับโครงสร้างของ SIP server แต่ละตัวด้วย ซึ่งหากจะทำการจำลองสถานการณ์โดยใช้เครื่องมือทดสอบหลายๆเครื่องก็เป็นไปได้ยากและไม่สะดวก และพวกเรายังปรับปรุงให้เป็นไปตามมาตรฐาน RFC3261 มากขึ้น รวมทั้งยังได้เพิ่ม scenario ที่เป็น xml file ขึ้นในลักษณะการ call ที่เป็น multiuser และให้ส่ง call request ผ่าน SIP proxy แทนที่จะส่งระหว่าง UAC และ UAS โดยตรงเท่านั้น

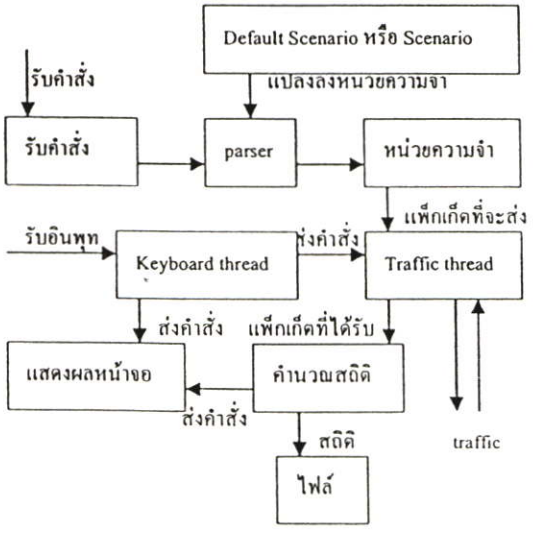
3.1 การพอร์ทโปรแกรมจากลีนุกซ์ขึ้นมาบนวินโดวส์

เราได้ทำการเปลี่ยนไลบรารี่ที่มีใช้งานอยู่เฉพาะบนลีนุกซ์ ให้เป็นไลบรารี่ที่มีอยู่บนวินโดวส์ คือ

1. ส่วนของ Socket เดิมบนลีนุกซ์มาเป็น winsock
2. ส่วนการทำงานเกี่ยวกับ regular expression ใช้ PCRE (Perl-compatible regular expression library)
3. สำหรับคำสั่งการทำงานที่มีอยู่บนลีนุกซ์เท่านั้นจะใช้ UnixEm ทดแทน ซึ่งเป็น Synesis Software UNIX Emulation for Win32

และได้ทำการแก้ไขเพิ่มเติม source code ที่ไม่สามารถหาไลบรารี่มาทดแทนได้ โดยการเพิ่มฟังก์ชั่น, Struct หรือ Class เข้าไปทดแทน รวมทั้งทำการเปลี่ยนคำสั่งต่างๆให้เหมาะสมที่จะนำมาใช้บนวินโดวส์

3.2 โครงสร้างที่สำคัญของโปรแกรม

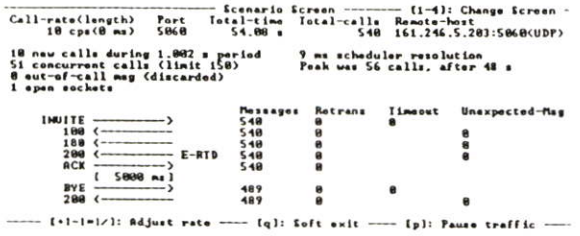


รูปที่ 2 ขั้นตอนการทำงานของโปรแกรม

1. ส่วนรับคำสั่ง – จะรับคำสั่งเข้ามาทาง argument ที่ต่อท้ายชื่อโปรแกรมทาง command prompt เช่น test –sf register.xml 161.246.5.203
2. ส่วน parser – จะอ่าน scenario หรือสถานการณ์จาก text file หรือจาก default scenario เองที่มีอยู่ภายในโปรแกรมมาเก็บไว้บนหน่วยความจำ เพื่อทำการรับส่งแพ็กเก็ตในลำดับต่อไป
3. ส่วน Keyboard thread - เป็นส่วนที่รับอินพุททางคีย์บอร์ดในขณะที่ทำการรันโปรแกรมอยู่เพื่อการแสดงผลที่ได้รับจากการทดสอบหรือเพื่อปรับแต่งอัตราการรับส่งต่างๆ
4. ส่วน Traffic thread – เป็นส่วนที่ควบคุมการรับส่งแพ็กเก็ตทั้งหมดที่เป็น SIP รวมทั้งปรับเปลี่ยนการรับส่งให้เป็นไปตาม scenario หรือสถานการณ์ที่สร้างขึ้น
5. ส่วนคำนวณสถิติ – จะทำการเก็บค่าหรือผลที่ได้รับจากการทดลองมาคำนวณแล้วเก็บไว้เพื่อแสดงผลที่หน้าจอต่อไป หรือทำการเก็บสถิติเพื่อ dump ลงไฟล์ และสามารถเก็บแพ็กเก็ตทั้งหมดที่เกิดขึ้นจากการรับส่งลงไฟล์ด้วย
6. ส่วนแสดงผล – แสดงผลที่ได้จากการคำนวณสถิติ หรือรูปแบบสถานการณ์ที่ใช้ทดสอบในขณะนั้นทางหน้าจอ

4. ผลการทดลองการใช้งาน SIP_CGA

เราได้ทำการทดสอบกับ SIP server ชื่อ VOCAL [9] โดยได้ทดลองส่งแพ็กเก็ต REGISTER และ INVITE ของผู้ใช้จำนวน 50 คน ด้วยอัตราจำนวนการ Call 10 ครั้งต่อวินาที ซึ่งหมายความว่าผู้ใช้คนหนึ่งจะถูก INVITE ทุกๆ 5 วินาทีหรือจะมีผู้ใช้ถูก INVITE วินาทีละ 10 คน โดยทำการจำลองส่งแพ็กเก็ตว่าแสมือนผู้ใช้แต่ละคนใช้ SIP phone ของ CISCO ซึ่งเหมาะสมกับการใช้ VOCAL โดยที่รูปแบบการทดสอบจะเป็นไปตามรูปที่ 1 คือทำการสร้างแพ็กเก็ตโดยใช้เครื่องหนึ่ง แล้วทำการ REGISTER ไปยัง VOCAL หลังจากนั้นจึง INVITE ผ่าน VOCAL ไปยังอีกเครื่องหนึ่งซึ่งทำการ REGISTER ไปที่ VOCAL เช่นเดียวกัน

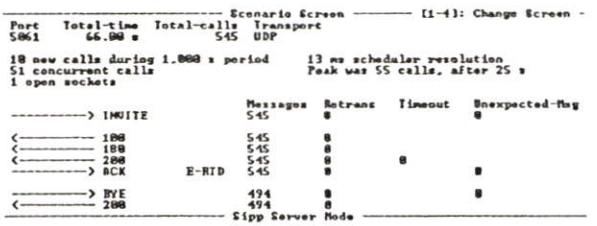


รูปที่ 3 หน้าจอแสดงผลการส่งของฝั่งที่ทำการ INVITE

Counter Name	Periodic value	Cumulative value
Elapsed Time	00:00:01:001	00:03:01:070
Call Rate	9.990 cps	9.996 cps
Incoming call created	0	0
OutGoing call created	10	1810
Total Call created		1810
Current Call	-103	
Successful call	7	1665
Failed call	0	248
Response Time	00:00:04:006	00:00:01:539
Call Length	00:00:12:288	00:00:07:711

รูปที่ 4 หน้าจอแสดงการสรุปผลการส่งของฝั่งที่ทำการ INVITE

สำหรับหน้าจอแสดงผลของฝั่งที่ถูก INVITE ก็จะเป็นในลักษณะเดียวกัน ขึ้นอยู่กับสถานการณ์ที่สร้างขึ้นให้ทั้ง 2 ฝั่งนั้นทำการโต้ตอบกันอย่างไร



รูปที่ 5 หน้าจอแสดงผลของฝั่งที่ถูก INVITE

จากรูปจะแสดงลักษณะการโต้ตอบกันระหว่างผู้ที่ทำการ INVITE และผู้ที่ถูก INVITE โดยที่ผลลัพธ์ของการทดสอบนั้นจะถูกแสดงออกมาในลักษณะ real-time โดยจะทำการรีเฟรชหน้าจอหรือผลลัพธ์ทุกๆ 1 วินาที

5. สรุป

บทความนี้นำเสนอถึงการพัฒนา SIP_CGA : SIP Call Generator and Analyzer โดยได้ทำการพัฒนาต่อจาก SIPp ซึ่งเป็น open source และจากผลการทดลองใช้งานพบว่ามีความสามารถในการสร้างแพ็กเก็ตของ SIP ให้มีความใกล้เคียงกับการใช้งานจริงตามมาตรฐาน RFC3261 ซึ่งไม่ได้ด้อยไปกว่าโปรแกรมอื่นๆเลยที่อยู่ทั่วๆไป ทั้งยังสามารถใช้งานได้บนระบบวินโดวส์อีกด้วย ซึ่งโดยทั่วๆไปแล้วนั้นยังไม่มีโปรแกรมประเภทนี้ที่เป็นโปรแกรม open source และยังได้เพิ่มความสามารถในการสร้างสถานการณ์จำลองเป็นผู้ใช้ SIP phone หลายๆคนได้ พร้อมทั้งมีความยืดหยุ่นในการใช้งานผ่านทางไฟล์ scenario file เพื่อเพิ่มเติมสถานการณ์เองภายหลังได้อีกด้วย ทั้งนี้หากได้รับการพัฒนาต่อให้สามารถใช้งานในรูปแบบที่มีการใช้งานและการแสดงผลเป็นแบบกราฟฟิค (GUI) ก็จะทำให้การแสดงผลเป็นไปอย่างง่ายได้และสื่อความเข้าใจได้มากยิ่งขึ้น รวมทั้งพัฒนาต่อให้สามารถครอบคลุมความสามารถให้เป็นไปตามมาตรฐานมากขึ้นก็จะดีมาก เพื่อการทดลองหาประสิทธิภาพการใช้งาน SIP server และ SIP device ต่างๆเมื่อมีผู้ใช้จำนวนมากจะได้มีความใกล้เคียงกับการใช้งานจริงมากที่สุด และปรับแต่งค่าพารามิเตอร์ต่างๆให้เหมาะสมตามความต้องการได้

เอกสารอ้างอิง

[1] RFC3261, "SIP : Session Initiation Protocol", June 2002

[2] 3GPP TSG SSA, IP Multimedia Subsystem (IMS) – Stage 2 (Release 5), TS 23.228 v. 5.4.1, 2002-04.

[3] 3GPP TSG CN, Signaling Flows for the IP Multimedia Call Control Based on SIP and SDP - Stage 3 (Release 5), TS 24.228 v. 2.0.2, 2002-03.

[4] 3GPP TSG CN, IP Multimedia Call Control Protocol Based on SIP and SDP - Stage 3 (Release 5), TS 24.229 v. 2.0.1, 2002-03.

[5] "SIPp", http://sipp.sourceforge.net/

[6] "SIPbomber", http://metalinkltd.com/eng/downloads/

[7] "Sipsak", http://sipsak.org/

[8] "SIPSTONE", http://www.sipstone.org/

[9] "VOCAL", http://www.vovida.org/

ประวัติผู้เขียนบทความ

นายวุฒินัย กาญจนศร จบการศึกษาระดับปริญญาตรี วิศวกรรมศาสตร์บัณฑิต สาขาวิศวกรรมคอมพิวเตอร์ จากสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ปัจจุบันกำลังศึกษาต่อในระดับปริญญาโท ในคณะวิศวกรรมศาสตร์ สาขาวิศวกรรมคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง สนใจในงานวิจัยด้าน SIP, Voice Over IP, Mobile and Wireless Networks และ Network

SIP_CGA : SIP Call Generator and Analyzer

Wutthinai Kanjanasorn and Surin Kittitornkun
Faculty of Engineering
King Mongkut's Institute of Technology Ladkrabang,
Email: wutthinai@yahoo.com, kksurin@kmitl.ac.th

SIP_CGA : SIP Call Generator and Analyzer

Wutthinai Kanjanasorn and Surin Kittitornkun
Faculty of Engineering King Mongkut's Institute of Technology Ladkrabang,
Ladkrabang, Bangkok 10520, Thailand
Email: wutthinai@yahoo.com, kksurin@kmitl.ac.th

Abstract

SIP Call Generator and Analyzer (SIP_CGA) is used for testing SIP server and analysing its performance. It can emulate almost every real situations. SIP_CGA is developed on SIPp, which is an opensource running on Linux system, and ported to run on Windows XP. In addition, SIP_CGA has been improved to comply with SIP standard and emulate multiuser situations that other programs can not. This paper explains how to use SIP_CGA, its specification by comparing with other similar programs, its structure and development.

Key Words: SIP, Call Generator, Analyzer, performance, SIPp

1. Introduction

From using VoIP (Voice Over IP) is more popular now. SIP (Session Initiation Protocol) RFC3261 [1] is an IETF application layer control protocol for creating, modifying and terminating sessions with one or more participants.

In the panorama of protocols for IP telephony, SIP is not the only possible choice. For instance, ITU-T H.323 is an alternative candidate. However, SIP has recently gained the increasing interest of universities, standardization organizations and companies. One of the main drivers for this fact has been the decision of 3GPP (Third Generation Partnership Program) that, in the year 2000, has selected SIP as call control protocol for 3G IPbased mobile networks (3GPP Release 5) [2], [3], [4]. The SIP protocol enables a wide range of applications ranging from Multimedia over IP to Instant Messaging, Presence, and Rich Calls.

From many more SIP users, this is necessary to have a testing performance tool for SIP server and network. This tool will be useful for setting SIP message parameter to support increased users and to increased each user's performance. This tool will analyze the problem that maybe happen in difference paths. For example, bottleneck in network. Many existent SIP Call Generator and Analyzer don't comply with RFC3261 standard and can not emulate multiuser. Furthermore they can not create and send SIP message from one to another user pass through SIP proxy correctly. Almost of them run on Unix System. So, we would have a SIP Call Generator and Analyzer can be run on Windows, can emulate multiuser , comply with standard, tester can create his/her situations and similar real using.

This paper is organized as follows. Section 2 shows related works and specification of SIP_CGA, our development of SIP_CGA is described in section 3. Experimental results are discussed and concluded in section 4 and 5, respectively.

2. Related works and specification of SIP_CGA

2.1 Related works

SIP Call Generator and Analyzer is the tool for create and send many packets of SIP message. Emulate large SIP phone users for testing performance of SIP server and network, analyse and show the result of generating many SIP packets in each situation.

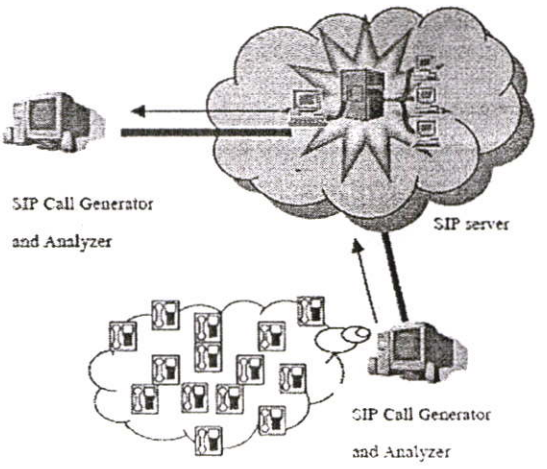


Figure 1. SIP Call Generator and Analyzer general work flow

The result of generating will be useful for setting the parameter of SIP message and identify to the problem. Example parameter is expiration ("Expire:" in SIP header) of REGISTER method, frequency of sending REGISTER packet for each user and the number of retransmission packet when REGISTER method is unsuccess.This figure show 2 SIP Call Generator and Analyzer have request and respond between them. Emulate the situation follow tester command. They communicate across network use SIP server is coordinator.

2.2 Specification of SIP_CGA

This comparison will show the specification between SIP_CGA and other programs. Those programs are

1. SIPp 1.0 (SP) [5]
2. Sipbomber 0.7 (SB) [6]
3. Sipsak 0.8.12 (SS) [7]

Table 1. SIP_CGA vs others

Specification	SP	SB	SS	Ours
Language	C/C++	Java	C++	C/C++
Platform	Linux	Redhat9	Linux	Windows
Generate many packet	✓	✓	✓	✓
Test between UAC and UAS	✓		✓	✓
Create situation	✓	✓		✓
Send packets through SIP proxy	Some packets	Some packets	Some packets	Some packets
Emulate Multiuser				✓
Transport layer protocol	TCP/ UDP	TCP/ UDP	UDP	TCP/UDP
GUI		✓		
Calculate Statistic	✓	✓		✓

SIP header format of each program still has problems now. Then sending SIP message from one UA to the another through SIP server can not do correctly. For instance, SIP header in Sipsak does not have ROUTE field then Sipsak can not send INVITE messages through SIP server but can send direct between UAC and UAS. Sipbomber has been developed for test the implementation of SIP server only. Not for test between UA.

3. Development of SIP_CGA

SIP_CGA has been developed from SIPp version 1.0 which is a C/C++ opensource program and run on Linux system in Redhat family in shell terminal. So we ported SIPp to run on WindowsXP in command prompt by using Microsoft Visual C++ version 6 service pack 5. And we also added other abilities to it for work comply with RFC3261 more too.

SIPp is the opensource that includes a few basic SIPSTONE -> user agent scenarios (UAC and UAS) and establishes and releases multiple calls with INVITE and BYE methods. SIPSTONE has developed by Columbia University and Ubiquity who are SIP creator and push SIP into standard.

We add ability to it for emulating SIP multiuser like many user have large usages instead of only one user have large usages. We develop it to work comply with RFC3261 more and add scenario XML files that emulate multiuser situation have calling through SIP proxy instead calling only between UAC and UAS (not pass SIP proxy).

3.1 Porting from Linux to Windows

We change Linux's libraries to Windows's libraries :

1. from unix SOCKET to WINSOCK.
2. related fields about Regular Expression use PCRE (Perl-Compatible Regular Expression library).
3. Use UnixEm (Synesis Software UNIX emulation for Win32) instead of unix C/C++ command that only have in unix.

And correct source code that can not find library for using instead, by add C/C++ function, Struct, Class and change its command for running on Windows correctly.

3.2 Main program structure

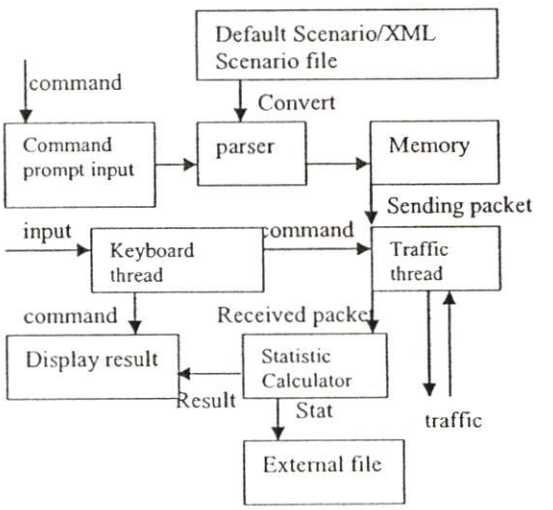


Figure 2. main program structure of SIP_CGA

1. Command prompt input – receive command from the argument after program name. Example : test –sf register.xml 161.246.5.203 means load scenario from register.xml and generate packet and send to SIP server locate at 161.246.5.203.
2. Parser – read the scenario from XML file or default scenario in self program and convert it into memory for generating and sending in next order.
3. Keyboard thread – receive command from keyboard when program is running for difference display result and for setting generating's parameter.
4. Traffic thread – control sending and receiving SIP messages. Adjust sending and receiving follow to created scenario.
5. Statistic Calculator – store the result from testing, calculate it and display on screen or dump stat to file. And can store all packets which are happened from generating too.
6. Display – Display all stat and traffic result from generating on screen.

4. Experimental results

We test SIP_CGA with VOCAL [9] as SIP server. We use SIP_CGA to emulate INVITE (call) situation between 50 SIP phones.

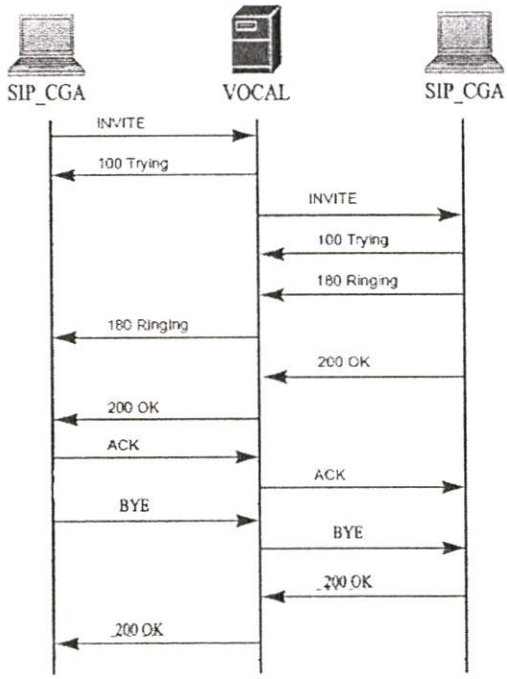


Figure 3. INVITE call flow between two SIP_CGAs INVITING and INVITED

First, we run SIP_CGA on PC to register as SIP users and run it again on another PC for register as another SIP users. Second, stop send REGISTER packets to VOCAL when register complete both 2 PCs and all users. Final, use SIP_CGA on first PC generate INVITE packets to SIP users which register on another PC. We generate INVITE message with rate 10 calls per second. So, one user will be invited every 5 seconds or in one second period will have 10 users be invited. Finally after register complete, SIP_CGA will wait for 5 seconds and send BYE message to close connection. Generated packet look like have been sent form CISCO SIP phone that appropriate with VOCAL. See figure 4 for INVITE situation packet flow.

This figure 3 show INVITE call flow after both 2 SIP_CGAs registered to VOCAL already complete. Our experiment base on this call flow.

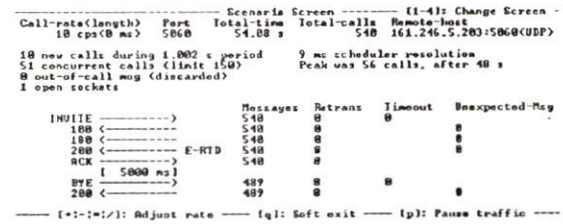


Figure 4. Status screen of SIP_CGA (INVITING)

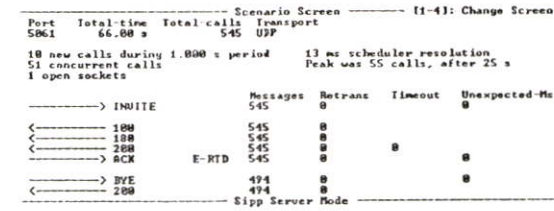


Figure 5. Status screen of SIP_CGA (INVITED)

Counter Name	Periodic value	Cumulative value
Elapsed Time	00:00:01:001	00:03:01:070
Call Rate	9.990 cps	9.996 cps
Incoming call created	0	0
OutGoing call created	10	1810
Total Call created		1810
Current Call	-103	
Successful call	7	1665
Failed call	0	240
Response Time	00:00:04:006	00:00:01:539
Call Length	00:00:12:288	00:00:07:711

Figure 6. Summary screen of SIP_CGA (INVITING)

For summary screen of INVITED side is same INVITING side. Up to scenario on both sides that how to request and response between them.

Figure 3, 5, 6 can show characteristic of request and response between inviter and invited by display on screen in real-time by refresh the result every second.

In Using SIP_CGA we found that the memory usage for PC which have run SIP_CGA will be increase by size of waiting message queue and increase by linear constant. It means that when SIP_CGA INVITING side generate and send packets to VOCAL or INVITED, and them do not reply with response packets in time, SIP_CGA which start INVITE will use memory more for waiting response packets that expect receive. Average 32 KBs per 10 waiting packets in INVITE situation.

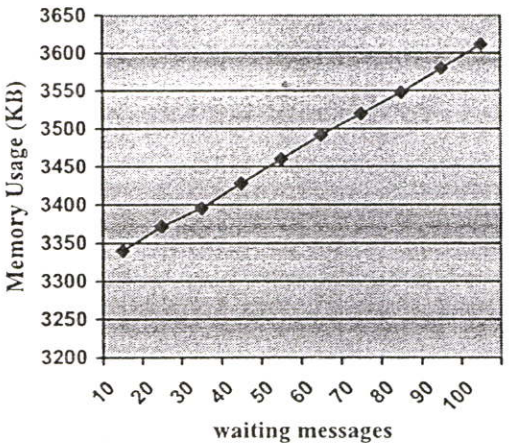


Figure 7. SIP_CGA memory usage vs number of waiting messages

5. Conclusion

This paper presents development of SIP_CGA : SIP Call Generator and Analyzer. SIP_CGA has developed from SIPp which is an opensorce. Refer to testing result found that SIP_CGA has ability to generate SIP packets nearly RFC3261 that not inferiorer than other programs and can run on Windows. SIP_CGA can emulate large multiuser as many SIP phones while other opensoruce programs can not. SIP_CGA has flexibility in usage, tester can create his/her situation by XML file. Also its memory usage is not a problem because increase by linear constant. If SIP_CGA is developed continually to work under GUI (Graffic User Interface), it will easy to use and understand more. Develop it to fully work comply with RFC3261 standard is very good for test performance of SIP server and network when have many users. SIP_CGA will can emulate SIP packets almost real usage and can adjust SIP message parameter more easily.

6. References

[1] RFC3261, "SIP : Session Initiation Protocol", June 2002

[2] 3GPP TSG SSA, IP Multimedia Subsystem (IMS) – Stage *2 (Release 5), TS 23.228 v. 5.4.1, 2002-04.*

[3] 3GPP TSG CN, Signaling Flows for the IP Multimedia Call Control Based on SIP and SDP - Stage 3 (Release 5), TS 24.228 v. 2.0.2, 2002-03.

[4] 3GPP TSG CN, IP Multimedia Call Control Protocol Based on SIP and SDP - Stage 3 (Release 5), TS 24.229v. 2.0.1, 2002-03.

[5] "SIPp", http://sipp.sourceforge.net/

[6] "SIPbomber", http://metalinkltd.com/eng/downloads/

[7] "Sipsak", http://sipsak.org/

[8] "SIPSTONE", http://www.sipstone.org/

[9] "VOCAL", http://www.vovida.org/

ประวัติผู้เขียน

ชื่อ-นามสกุล	นายวุฒินัย กาญจนศร
ประวัติการศึกษา	สำเร็จการศึกษาระดับปริญญาตรี หลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์ จากคณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ปีการศึกษา 2546 และเข้าศึกษาต่อในระดับปริญญาโทที่สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง หลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ในปีการศึกษา 2547
งานวิจัยที่สนใจ	SIP, Voice Over IP, Mobile and Wireless Networks และ Network