ระบบบันทึกบริการถ่ายทอดการสื่อสารด้วยสถาปัตยกรรมตรวจจับแพ็คเก็ตหลายถัง

Video Relay Service Recording System Using Packet-Capture Multiple Buckets Architecture

พรพิมล รามัญอุดม[1,*] และ บุญชัย งามวงศ์วัฒนา[1]

[1]ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ สำนักงานพัฒนาวิทยาศาสตร์และเทคโนโลยีแห่งชาติ คลองหนึ่ง คลองหลวง ปทุมธานี 12120

Pornpimol Ramonudom[1,*] and Boonchai Ngamwongwattana[1]

[1]National Electronics and Computer Technology Center, National Science and Technology Development Agency, Khlong Nueng, Khlong Luang, Pathum Thani 12120, Thailand

***Corresponding Author E-mail:** pornpimol.ramonudom@nectec.or.th

Received: Sep 05, 2022; **Revised:** Mar 23, 2023; **Accepted:** May 22, 2023

บทคัดย่อ

บทความนี้นำเสนองานวิจัยพัฒนาระบบบันทึกบริการอัตโนมัติสำหรับแพลตฟอร์มบริการถ่ายทอดการสื่อสาร ซึ่งเป็นโครงการความร่วมมือระหว่างสำนักงานพัฒนาวิทยาศาสตร์และเทคโนโลยีแห่งชาติ (สวทช.) และศูนย์บริการถ่ายทอดการสื่อสารแห่งประเทศไทย (TTRS - Thai Telecommunication Relay Service) เพื่อจัดเก็บและบันทึกบริการถ่ายทอดการสื่อสารแบบ Video Relay Service สำหรับผู้พิการทางการได้ยินที่ไม่สามารถสื่อสารด้วยเสียง โดยนำองค์ความรู้ทางด้าน Network-Based Monitoring มาพัฒนาระบบให้สามารถทำงานได้อย่างอัตโนมัติ (Fully Automated) และมีความฉลาด (Intelligence) ในการวิเคราะห์ขั้นสูงเกี่ยวกับรูปแบบต่างๆ ของบริการสนทนา การทำงานโดยย่อระบบจะทำหน้าที่ในการตรวจจับ Network Traffic และวิเคราะห์ Network Packet, Signaling และ Protocol ต่างๆ ที่เกี่ยวข้อง เช่น TCP, UDP, SIP, SDP, RTP รวมถึง H.264, G.711 Encoding และทำการบันทึก Media Stream ต่างๆ ของการสนทนา แล้วสร้างเป็นไฟล์วิดีโอที่บันทึกบริการทั้งหมด และส่งไปเก็บที่ Cloud Storage เพื่อการใช้ประโยชน์ต่อไป เช่น การตรวจสอบกรณีร้องเรียน การประเมินการบริการ การปรับปรุงและเพิ่มประสิทธิภาพการบริการให้ดียิ่งขึ้น ความท้าทายของการพัฒนาระบบนี้คือความสามารถในการจัดการและประมวล Real-Time Interactive Multi Flows เนื่องจากระบบจะต้องตรวจจับและประมวลผล Call Signaling และ Media Stream จำนวนมากที่เกิดขึ้นแบบ Interactive Real Time กระบวนการ Optimization จึงมีความสำคัญอย่างยิ่ง เพื่อทำให้การประมวลผลสามารถทำได้อย่างรวดเร็ว ทันต่อเวลา และไม่เกิดปัญหา Buffer Overflow ซึ่งจะมีผลต่อคุณภาพของการบันทึกบริการ โดยระบบที่พัฒนาขึ้นได้มีการออกแบบสถาปัตยกรรมการประมวลผล PCAP (Packet Capture) แบบใหม่เรียกว่า Multiple Buckets Architecture ซึ่งผลการทดสอบแสดงให้เห็นว่าสามารถลดโอกาสเกิด Buffer Overflow ได้เป็นอย่างมาก และสามารถทำงานได้อย่างมีประสิทธิภาพมากกว่า เมื่อเปรียบเทียบกับการประมวลผลแบบทั่วไปที่เป็นแบบ Single Bucket

คำสำคัญ: การตรวจจับเครือข่าย, การตรวจจับการรับส่งข้อมูลเครือข่าย, การวิเคราะห์สัญญาณ SIP, โปรโตคอล RTP, ไฟล์บันทึกบริการ, สถาปัตยกรรมการตรวจจับแพ็คเก็ต

Abstract

This paper presents a research and development work for the automatic service recording system using network-based monitoring approach. The project is under the cooperation of National Science and Technology Development Agency (NSTDA) and Thai Telecommunication Relay Service (TTRS). The goal is to archive and record video relay services offered to the deaf and hard of hearing community. Using the network-based monitoring approach, we can develop a video recording system that is fully automated and intelligent to analyze different kinds of video relay services. In brief, the system functions to capture network traffic and analyze network packets, signaling, and related protocols such as TCP, UDP, SIP, SDP, RTP, as well as H.264, G.711 encoding. Then, it records the relevant media streams, creates a video file containing the whole video relay service conversation, and send the file to archive in the cloud storage. The benefits include verification in case of appeal, service evaluation, and service improvement. A technical challenge in this system development is how to process real-time interactive multi flows because it must detect and process a large number of call signaling and media streams that occur as interactive real time. The optimization is vital so that the system can swiftly do the processing in time, without causing the buffer overflow problem that can degrade the recording quality. In this regard, we developed a novel methodology for processing PCAP (Packet Capture) called Multiple Buckets Architecture. Our test results showed that it can significantly reduce the probability of buffer overflow, making the processing more efficient when compared to the traditional processing of single bucket.

Keywords: Network Monitoring, Packet Capture, SIP Signaling, RTP Protocol, Video Streaming, Packet Capture Architecture

1. บทนำ

ศูนย์บริการถ่ายทอดการสื่อสารแห่งประเทศไทย (Thailand Telecommunication Relay Service หรือ TTRS) ให้บริการถ่ายทอดการสื่อสารแบบ Video Relay Service สำหรับคนหูหนวกและผู้พิการทางการพูด มีความจำเป็นต้องจัดเก็บบันทึกบริการการสนทนาวิดีโอ เพื่อให้สามารถสืบค้นย้อนหลังได้ เนื่องจากต้องใช้ตรวจสอบในกรณีร้องเรียน การประเมินการบริการ และการปรับปรุงเพิ่มประสิทธิภาพการบริการให้ดียิ่งขึ้น อย่างไรก็ตาม การใช้เครื่องมือทั่วไป (เช่น Video Screen Capture) เพื่อบันทึกการสนทนาวิดีโอเป็นไฟล์บันทึกบริการ ยังมีปัญหาในกระบวนการ Post Processing ขั้นตอนนี้มีความยุ่งยาก ใช้เวลาประมวลผลนาน ไม่มีประสิทธิภาพ และกระทบต่อการทำงานของเจ้าหน้าที่อีกด้วย การพัฒนาระบบจัดเก็บและบันทึกบริการด้วยงานวิจัยทางด้าน Network-Based Monitoring ทำให้ระบบสามารถตรวจจับ Network Traffic และวิเคราะห์ Call Signaling และ Protocol ต่างๆ ที่เกี่ยวข้อง และมีความฉลาด (Intelligence) ในการวิเคราะห์ขั้นสูงเกี่ยวกับรูปแบบต่างๆ ของบริการสนทนาได้ ดังนั้นจึงสามารถทำงานได้อย่างอัตโนมัติ (Fully Automated) ในการตรวจจับสายเรียกเข้า การเริ่มและหยุดบันทึก รวมถึงการประมวลผล Media Stream ต่างๆ เพื่อรวมเป็นไฟล์ที่บันทึกบริการทั้งหมด นอกจากนี้ระบบจะส่งไฟล์ไปเก็บที่ Cloud Storage อย่างอัตโนมัติด้วย โดยเจ้าหน้าที่สามารถสืบค้นไฟล์บันทึกบริการได้อย่างง่ายดายผ่าน Web Portal ของระบบ ดังนั้นระบบที่พัฒนาขึ้นจึงสามารถแก้ปัญหาการบันทึกบริการและการจัดเก็บรวบรวมข้อมูล และเพิ่มประสิทธิภาพการบริการของศูนย์บริการถ่ายทอดการสื่อสารแห่งประเทศไทยได้เป็นอย่างมาก

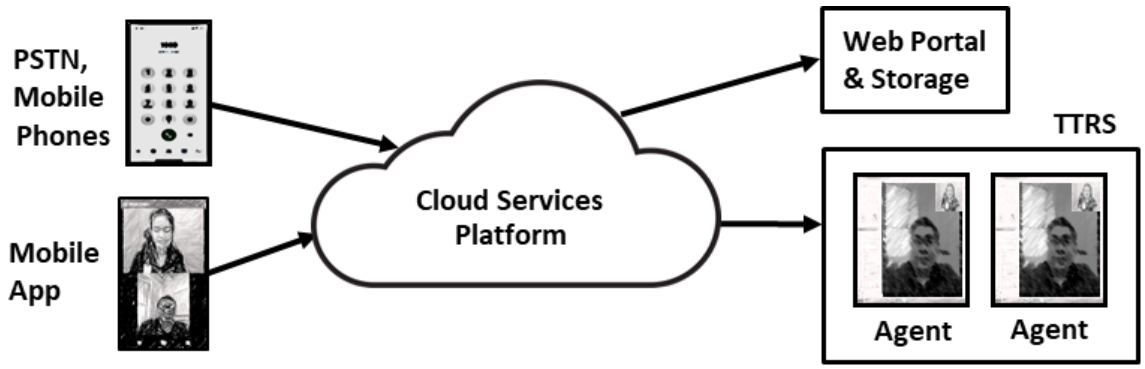
2. วรรณกรรมที่เกี่ยวข้อง

PCAP (Packet Capture) [1],[2] เป็น API สำหรับการตรวจจับ Network Traffic ในระบบเครือข่าย ที่มีการใช้งานกันอย่างแพร่หลาย เริ่มต้นถูกพัฒนาที่ Lawrence Berkeley Laboratory ในรูปแบบของ Libpcap บนเครื่องคอมพิวเตอร์ในระบบปฏิบัติการ Unix จนถึงปัจจุบัน PCAP API ถือเป็น De-Facto Standard เพื่อใช้ในการตรวจจับ Network Traffic เพื่อบันทึกการสื่อสารบนระบบเครือข่ายและการนำไปประมวลผลต่อไป PCAP API ได้ถูกนำไปพัฒนาเป็นโปรแกรมประยุกต์ทางด้านระบบเครือข่ายมากมาย เช่น Protocol Analyzer, Network Monitoring, IDS (Intrusion Detection System), Network Forensic Tool เป็นต้น ความท้าทายในการพัฒนาระบบด้วย PCAP API ที่สำคัญคือ เมื่อจำเป็นที่จะต้องวิเคราะห์ประมวลผลขั้นสูงหรือในปริมาณมากๆ สำหรับ Network Traffic ที่เข้ามาแบบ Real Time อาจทำให้เกิดปัญหา Buffer Overflow และเกิดการสูญเสีย Packets ไปบางส่วนได้ จากวรรณกรรมที่เกี่ยวข้องได้มีการอ้างถึงประเด็นปัญหานี้เช่นกัน เนื่องจาก PCAP API รองรับการทำงานแบบ Single Thread เท่านั้น ระบบต่างๆ ส่วนใหญ่จึงยังคงทำงานเป็นแบบ Single Thread โดย N. Bonelli, et al. [3] ได้พัฒนาต่อยอด PCAP API บน Linux เพื่อเพิ่ม Packet Fan-Out ให้สามารถแจกจ่าย Traffic Workload เพื่อการประมวลผลแบบ Multi-Core Processing ได้ J. F. Zazo, et al. [4] นำเสนอวิธีการประมวลแบบ Hardware-Based ด้วย FPGA เพื่อเพิ่มความเร็วในการประมวลผลสำหรับการตรวจจับ Network Traffic ปริมาณมากที่ความเร็วมากกว่า 1 Tb/s นอกจากนี้มีวรรณกรรมที่เกี่ยวเนื่องกับการเพิ่มประสิทธิภาพการตรวจจับ Network Traffic แนวทางอื่นๆ เช่น V. Duarte, et al. [5] นำเสนอการทำงานในแบบ Distributed Monitoring Tool และ J. Liu, et al. [6] เสนอวิธีการ Traffic Monitoring บนระบบปฏิบัติการ iOS

สำหรับงานวิจัยในบทความนี้มีจุดมุ่งหมายที่จะต้องทำงานร่วมกับ PCAP Driver บน Windows เวอร์ชันต่างๆ ที่ใช้งานอย่างแพร่หลายได้ เช่น WinPcap [7], Win10Pcap [8], และ Npcap [9] และด้วยข้อจำกัดของ Capture Buffer ที่ 2 MB และ PCAP API ไม่มีทางเลือกในการปรับ Buffer Size [10] จึงได้มีการพัฒนาสถาปัตยกรรมแบบ Multiple Buckets เพื่อให้สามารถประมวลผล Real-Time Interactive Multi Flows สำหรับการบันทึกการสนทนาบริการได้อย่างรวดเร็ว มีประสิทธิภาพ และไม่เกิดปัญหา Buffer Overflow **ตารางที่ 1** แสดงข้อมูลจำแนกวิธีการจัดการและประมวล Network Traffic แบบต่างๆ และเปรียบเทียบกับงานวิจัยที่เกี่ยวข้องก่อนหน้า

ตารางที่ 1 จำแนกวิธีการจัดการและประมวล Network Traffic แบบต่างๆ

การจัดการ Traffic	หลักการทำงาน	สภาพแวดล้อมและการใช้งาน
Packet Fan-Out (N. Bonelli, et al. [3])	พัฒนา PCAP Library เพื่อเพิ่ม Packet Fan-Out สำหรับการประมวลผลแบบ Multi-Core Processing	การตรวจจับและประมวล Traffic ที่ความเร็วสูงระดับ Multi-Gigabit Interface สามารถประมวลผล Traffic ในระดับ Flow หรือแบบ Stateless ได้
TNT10G (J. F. Zazo, et al. [4])	ใช้การประมวลผลแบบ Hardware-Based ด้วย FPGA เพื่อเพิ่มความเร็วในการประมวลผล	การตรวจจับและประมวล Traffic ปริมาณมากที่ความเร็วมากกว่า 1 Tb/s เหมาะกับการประมวลผล Traffic แบบ Stateless หรือ Post Processing
Multiple Buckets (ระบบที่พัฒนาขึ้น)	ทำงานแบบ Multi-Thread ในระดับ Application เพื่อรองรับหลาย PCAP Instance ที่จำเป็นต้องใช้งาน	การตรวจจับและประมวล Traffic ที่ความเร็วปกติไม่เกิน 1 Gb/s สามารถประมวลผล Traffic และวิเคราะห์ได้ถึงระดับ Call ที่เป็น Interactive Multi Flows



รูปที่ 1 ภาพรวมการทำงานของระบบบันทึกบริการอัตโนมัติ

3. การวิจัยพัฒนาระบบ

ภาพรวมของบริการถ่ายทอดการสื่อสารแบบ Video Relay Service และระบบบันทึกบริการอัตโนมัติ แสดงดัง **รูปที่ 1** ผู้พิการทางการได้ยินสามารถใช้ Mobile App เพื่อติดต่อแบบวิดีโอด้วยภาษามือมายังเจ้าหน้าที่บริการถ่ายทอดการสื่อสารที่ศูนย์ TTRS เพื่อให้ช่วยโทรติดต่อสื่อสารกับบุคคลธรรมดาได้ โดยระบบบันทึกบริการอัตโนมัติจะถูกติดตั้งที่เครื่องคอมพิวเตอร์ของเจ้าหน้าที่ และจะทำการตรวจจับสายโทรเข้า-ออก และบันทึกบริการอย่างอัตโนมัติ และจะส่งไฟล์วิดีโอบันทึกบริการไปเก็บที่ Cloud Storage เจ้าหน้าที่สามารถสืบค้นข้อมูลการบริการย้อนหลังได้ผ่าน Web Portal เพื่อตรวจสอบในกรณีร้องเรียน การประเมินการบริการ และการปรับปรุงเพิ่มประสิทธิภาพการบริการให้ดียิ่งขึ้น โครงสร้างการทำงานของระบบบันทึกบริการอัตโนมัติ แสดงดัง**รูปที่ 2 และ 3** ประกอบด้วยโมดูลหลักๆ ดังนี้

3.1 Packet Capture Module

พัฒนาด้วย PCAP API ทำหน้าที่ตรวจจับ Packet ผ่าน Network Interface โดยจะทำการดึง Packet จาก Buffer เพื่อประมวลผลต่อไป จาก**รูปที่ 2 และ 3** เส้นทึบแสดง Flow ของการประมวลผล Packet

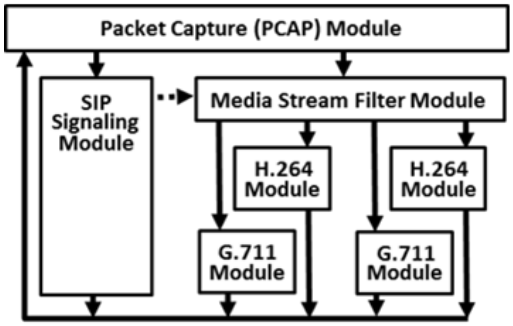
3.2 SIP Signaling Module

เป็นโมดูลหลัก ทำหน้าที่ในการตรวจจับและวิเคราะห์การสนทนาบริการที่เกิดขึ้น มีการทำงานดังนี้

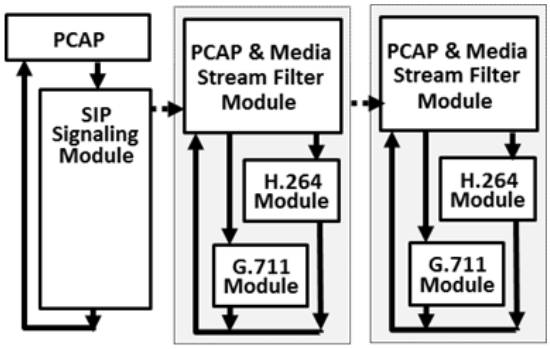
- **การวิเคราะห์สัญญาณ Call Signaling** ทำหน้าที่วิเคราะห์ SIP Signaling [11] เช่น INVITE, ACK, OK, BYE เป็นต้น รวมถึงวิเคราะห์ SDP (Session Description Protocol) [12] ซึ่งจะได้ข้อมูลและรายละเอียดเกี่ยวกับการติดต่อสื่อสาร เช่น Call-ID, IP Address, RTP (Real-Time Transport Protocol) Port, Contact Name, Video Encoding (เช่น H.264), Audio Encoding (เช่น G.711 PCMU, PCMA) เป็นต้น ข้อมูลเหล่านี้จะถูกส่งไปยัง Media Stream Filter Module เพื่อคัดกรอง Media Stream ต่อไป (จาก**รูปที่ 2 และ 3** แสดงด้วยเส้นประ)
- **การวิเคราะห์ Call Relay Service** เป็นการวิเคราะห์ขั้นสูงเกี่ยวกับการสนทนาบริการรูปแบบต่างๆ ซึ่งอาจจะประกอบด้วยหลายการสนทนาย่อยๆ เช่น ผู้พิการทางการได้ยินติดต่อผ่านเจ้าหน้าที่ เพื่อขอติดต่อไปยังบุคคลธรรมดา หรือการสนทนา Video Conference เพื่อเชื่อมการสนทนาวิดีโอ 3 สาย เป็นต้น โดยจะมีการเก็บข้อมูล Real Time ของความสัมพันธ์ของสายเรียกเข้าและโทรออก ข้อมูลเหล่านี้ จะใช้ในกระบวนการสร้างไฟล์บันทึกบริการต่อไป

3.3 Media Stream Filter Module

จะรับข้อมูลจาก SIP Signaling Module เกี่ยวกับ Media Stream ต่างๆ ที่จะทำการบันทึก แล้วทำหน้าที่ในการกรอง Packet ที่ต้องการ เพื่อส่งต่อไปยัง H.264 Module และ G.711 Module จาก**รูปที่ 2 และ 3** แสดงการทำงานซึ่งประกอบด้วย คู่ของ Video/Audio Stream ทางด้านขาไปและอีกคู่สำหรับทางด้านขากลับ



รูปที่ 2 โครงสร้างการทำงานแบบ Single Bucket



รูปที่ 3 โครงสร้างการทำงานแบบ Multiple Buckets

3.4 H.264 Module

ทำหน้าที่วิเคราะห์ RTP Packet [13] ที่เป็น H.264 Payload [14] แล้วบันทึก Video Stream เป็นไฟล์ Raw H.264 ประกอบด้วยการทำงานหลักๆ ดังนี้

- **การสร้าง RTP Packet Buffer** โดยใช้ค่า Sequence Number ใน RTP Header เพื่อตรวจสอบลำดับของ RTP Packet รวมถึงความผิดปกติของการลำดับผิด (Out-of-Order Packet) ซึ่งสามารถเกิดขึ้นได้ในการส่ง Packet ผ่านระบบเครือข่าย การสร้าง Buffer นี้จะช่วยแก้ไขความผิดปกติ เพื่อจัดลำดับให้ถูกต้อง (Packet Reorder) ก่อนการประมวลผลต่อไป
- **การวิเคราะห์ RTP Header** โดยปกติ Video Frame จะมีขนาดใหญ่มาก การส่ง Video Stream ผ่านระบบเครือข่าย Video Frame จะถูกส่งแยกออกจากกันเป็นหลาย Packet ต่อเนื่องกัน การตรวจสอบในส่วนของ Marker Bit เป็นตัวกำหนดว่า Packet นั้นเป็นส่วนสุดท้ายของ Video Frame ส่วนของ Timestamp เป็นค่าเวลาที่ Frame นั้นถูกสร้างขึ้น ดังนั้นหลายๆ Packet สามารถมีค่า Timestamp เดียวกันได้ ถ้าถูกแบ่งออกมาจาก Frame เดียวกัน ในส่วนของ Sequence Number จะใช้สำหรับการตรวจสอบการเรียงลำดับของ Packet รวมถึงความผิดปกติต่างๆ เช่น การสูญหาย การลำดับผิด เป็นต้น จากการวิเคราะห์ข้อมูลเหล่านี้ จะทำให้สามารถรวบรวมข้อมูลจากแพ็คเก็ตต่างๆ เข้าเป็น Video Frame ดังเดิมได้
- **การวิเคราะห์และบันทึก Raw H.264** โดยปกติ Video Stream จะมีการส่งเป็นแบบ Scalable Video Coding [15] (SVC) แบบ Temporal VFR (Variable Frame Rate) โดยที่ Presentation Time ของแต่ละ Video Frame จะไม่แน่นอน ขึ้นอยู่กับการเคลื่อนไหวในภาพวิดีโอ แต่การบันทึก Video Stream เป็นไฟล์ Raw H.264 ใน Container Format เช่น MP4 จะเป็นแบบ CFR (Constant Frame Rate) ดังนั้น จึงได้มีการพัฒนาวิธีการ VFR-to-CFR Mapping เพื่อให้สามารถรักษา Real-Time Presentation ได้อย่างถูกต้องในการบันทึกไฟล์วิดีโอ นอกจากนี้ วิธีการนี้ยังสามารถคำนวณและชดเชย Presentation Time ในกรณีที่ Video Frame อาจสูญเสียไปผ่านระบบเครือข่ายด้วย

3.5 G.711 Module

ทำหน้าที่วิเคราะห์ RTP Packet ที่เป็น G.711 Payload [16] แล้วบันทึก Audio Stream เป็นไฟล์ Raw G.711 ประกอบด้วยการทำงานหลักๆ ดังนี้

- **การสร้าง RTP Packet Buffer** เช่นเดียวกับกรณีของ H.264 Module โดยใช้ค่า Sequence Number ใน RTP Header เพื่อตรวจสอบลำดับของ RTP Packet รวมถึงความผิดปกติของการลำดับผิด (Out-of-Order Packet) การสร้าง Buffer นี้จะช่วยแก้ไขเพื่อจัดลำดับให้ถูกต้อง (Packet Reorder) ก่อนการประมวลผลต่อไป
- **การวิเคราะห์ RTP Header และบันทึก Raw G.711** ปกติ Audio Frame จะมีการส่งเป็นแบบ Constant Frame Rate โดย Presentation Time ของแต่ละ Audio Frame จะมีเวลาเท่ากัน กระบวนการวิเคราะห์ประกอบด้วย การจัดการ Packetization [17] และ DTX (Discontinuous Transmission) [18] และกรณีที่ Audio Frame อาจจะ

สูญเสียไปผ่านระบบเครือข่าย ซึ่งจะต้องมีการคำนวณและชดเชย Presentation Time ได้อย่างถูกต้องด้วย

3.6 Media File Creation Module

เมื่อ SIP Signaling Module ตรวจจับการสนทนาบริการจบลง จะมีการส่งสัญญาณมายังโมดูลนี้ พร้อมกับข้อมูลที่เกี่ยวข้อง เช่น ไฟล์ Raw Video/Audio ที่บันทึกได้ และข้อมูล Real Time ซึ่งแสดงความสัมพันธ์ของสายเรียกเข้า-ออก และ Media Stream ต่างๆ ที่เกิดขึ้น โดยมีการทำงานหลักๆ ดังนี้

- **การสร้างไฟล์บันทึกบริการ** ทำหน้าที่ในการประมวลผลไฟล์ Raw Video/Audio ที่บันทึกได้ และรวมเข้าด้วยกัน เพื่อสร้างเป็นไฟล์วิดีโอในรูปแบบ MP4 ที่บันทึกการสนทนาบริการทั้งหมด กระบวนการนี้ทำโดยใช้ FFmpeg Library [19] ประกอบด้วยการทำ Stream Merge, Concatenation จากไฟล์ Raw Video/Audio ที่บันทึกได้ โดยส่วนสำคัญคือ Stream Synchronization ซึ่งอ้างอิงจากข้อมูล Real Time เนื่องจาก ถ้ามีความคลาดเคลื่อน จะทำให้เกิดปัญหาต่างๆ ตามมาได้ เช่น Lip Sync ของ Video/Audio ไม่ตรงกัน
- **Queue Manager** ทำหน้าที่จัดลำดับการสร้างไฟล์บันทึกบริการแบบ FIFO (First-In, First-Out) เนื่องจากการประมวลผลไฟล์ Raw Video/Audio จำเป็นต้องใช้ทรัพยากรเครื่องสูง การประมวลผลโดยไม่มีการควบคุมการใช้ทรัพยากร จะทำให้เกิดปัญหา CPU, RAM Overload ซึ่งอาจจะมีผลกระทบต่อระบบที่กำลังบันทึก Media Stream อยู่

3.7 Cloud Service Module

ทำหน้าที่ในการเชื่อมต่อกับแพลตฟอร์มผ่าน REST API Web Services โดยจะมีการส่งข้อมูลบริการและสถานะต่างๆไปที่แพลตฟอร์มแบบ Real Time เช่น ชื่อผู้ติดต่อ เจ้าหน้าที่บริการ วันที่และเวลา รูปแบบบริการ เป็นต้น เมื่อการสนทนาบริการสิ้นสุดลง จะมีการส่งไฟล์บันทึกบริการไปเก็บที่ Cloud Storage อย่างอัตโนมัติ โดยเจ้าหน้าที่สามารถเข้าสู่ระบบ Web Portal เพื่อสืบค้นข้อมูลบริการ และเรียกดูไฟล์บันทึกบริการที่ต้องการได้

4. Multiple Buckets Architecture

ปัญหา PCAP Buffer Overflow สามารถอธิบายได้จาก **รูปที่ 2** โดยที่ PCAP Module จะตรวจจับ Packet ที่ผ่านเข้า-ออกจาก Network Interface ตลอดเวลา แล้วทำการดึง Packet ออกจาก Buffer ส่วนของ Packet ที่ไม่เกี่ยวข้องกับการบันทึกบริการจะไม่ผ่าน Media Stream Filter Module และไม่มีการประมวลผลใดๆ ส่วนของ Packet ที่เกี่ยวข้องกับ Media Stream จะถูกนำไปประมวลผล ซึ่งจะมีผลต่อ PCAP Buffer ด้วย สมการที่ (1) แสดงความสัมพันธ์ของ PCAP Buffer Utilization (b) ซึ่งเท่ากับส่วนต่างระหว่างอัตราเร็วของ Packet ที่เก็บเข้าไปใน Buffer ลบด้วยอัตราเร็วในการการดึง Packet ออกจาก Buffer แล้วคูณด้วยระยะเวลาของการบันทึกบริการ (t) โดย i = 1, 2, …, n ขึ้นอยู่กับรูปแบบของบริการ โดยกำหนดให้ r_{H264}, r_{G711} เป็นอัตราเร็วเฉลี่ยของ Stream H.264, G.711 และ p_{H264}, p_{G711} เป็นอัตราการประมวลผลเฉลี่ยของ Stream H.264, G.711 ตามลำดับ ในหน่วยไบต์ต่อวินาที

$$b = \left(\sum_{i=1}^{n}\left(r_{H264}^{i} + r_{G711}^{i}\right) - \sum_{i=1}^{n}\left(\left(r_{H264}^{i} - p_{H264}^{i}\right) + \left(r_{G711}^{i} - p_{G711}^{i}\right)\right)\right)(t) \quad (1)$$

จากสมการที่ (1) PCAP Buffer Utilization จึงขึ้นอยู่กับอัตราการประมวลผลของ Stream H.264, G.711 และระยะเวลาของการบันทึกบริการ ดังแสดงในสมการที่ (2)

$$b = t\sum_{i=1}^{n}\left(p_{H264}^{i} + p_{G711}^{i}\right) \quad (2)$$

ทั้งนี้ อัตราการประมวลผลของ Stream H.264, G.711 ยังขึ้นกับปัจจัยอื่นๆ ด้วย เช่น Media Packetization, Video Resolution, ทรัพยากรของเครื่องคอมพิวเตอร์ (CPU, RAM) ขณะนั้น เป็นต้น

การออกแบบสถาปัตยกรรมแบบใหม่ที่นำเสนอในงานวิจัยนี้เรียกว่า Multiple Buckets แสดงดัง**รูปที่ 3** โดยจะมีการสร้าง Thread เพื่อรองรับหลาย PCAP Instance ที่สามารถทำงานได้พร้อมๆ กัน เท่าที่จำเป็นต้องใช้งาน ตัวอย่างเช่น การสนทนาแบบ Video Call จะมี PCAP Instance หนึ่งเพื่อจัดการ H.264, G.711 Stream ที่ส่งมาจากผู้ติดต่อ และอีก

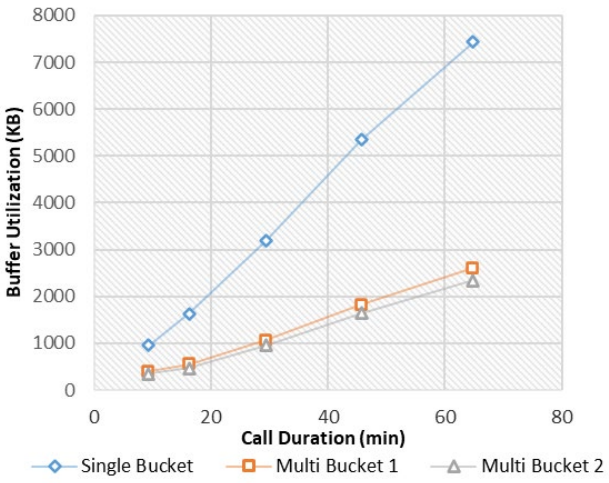
PCAP Instance หนึ่งเพื่อจัดการ H.264, G.711 Stream ที่ส่งกลับไปยังผู้ติดต่อ ข้อดีของ Multiple Buckets Architecture คือสามารถแก้ปัญหาและลดโอกาสเกิด PCAP Buffer Overflow ได้อย่างมีประสิทธิภาพมากกว่าการทำงานแบบทั่วไปซึ่งเป็นSingle Bucket Architecture

5. ผลการทดสอบประสิทธิภาพ

ตารางที่ 2 และ **รูปที่ 4** แสดงผลการทดสอบในสภาวะควบคุมเพื่อเปรียบเทียบ Buffer Utilization ระหว่าง Single Bucket กับ Multiple Buckets Architecture สำหรับเวลาการสนทนาบริการ (Call Duration) ที่แตกต่างกัน โดย Input Rate, Output Rate (KB/second) ในหน่วยกิโลไบต์ต่อวินาที เป็นอัตราเร็วเฉลี่ยของการเก็บ Packet เข้าไปใน Buffer และการดึงออกจาก Buffer ตามลำดับ และ Buffer Utilization (KB) ในหน่วยกิโลไบต์ แสดงถึง Buffer ที่จำเป็นต้องใช้เพื่อบันทึกการสนทนาบริการนั้น ๆ จากผลการทดสอบ แสดงให้เห็นว่าการใช้ 2 Buckets ทำให้ Buffer Utilization ลดลงได้กว่า 2/3 หรือประมาณ 65% เมื่อเปรียบเทียบกับการทำงานแบบ Single Bucket ดังที่แสดงในค่า Improved Buffer Utilization (%) ซึ่งหมายถึงเปอร์เซ็นต์การใช้งาน Buffer ที่ได้รับการพัฒนาให้สูงขึ้น ดังนั้น จึงสามารถลดโอกาสเกิดปัญหา PCAP Buffer Overflow ได้อย่างมาก นอกจากนี้ ยังทำให้ระบบสามารถบันทึกบริการสนทนาที่มีระยะเวลานานขึ้นได้อีกด้วย

ตารางที่ 2 ผลการทดสอบประสิทธิภาพระหว่าง Single Bucket กับ Multiple Buckets Architecture

	Single Bucket			Multiple Buckets							
				Bucket 1				Bucket 2			
Call Duration (min)	Input Rate (KB/s)	Output Rate (KB/s)	Buffer Util. (KB)	Input Rate (KB/s)	Output Rate (KB/s)	Buffer Util. (KB)	Imp. Buffer Util. (%)	Input Rate (KB/s)	Output Rate (KB/s)	Buffer Util. (KB)	Imp. Buffer Util. (%)
9.30	67.80	66.08	961.43	32.97	32.27	392.83	59.14	34.83	34.20	350.36	63.55
16.29	70.15	68.48	1,630.26	31.80	31.24	547.70	66.40	38.35	37.87	469.55	71.19
29.40	74.99	73.19	3,189.74	35.87	35.26	1,064.72	66.62	39.13	38.59	950.49	70.20
45.76	77.01	75.07	5,345.86	36.70	36.03	1,819.05	65.97	40.32	39.72	1,639.03	69.34
64.78	75.01	73.10	7,433.53	35.70	35.03	2,606.67	64.93	39.31	38.71	2,338.78	68.53



รูปที่ 4 กราฟแสดงผลการทดสอบประสิทธิภาพระหว่าง Single Bucket กับ Multiple Buckets Architecture

6. สรุป

บทความนี้ได้นำเสนองานวิจัยพัฒนาระบบบันทึกบริการถ่ายทอดการสื่อสารด้วยสถาปัตยกรรมตรวจจับแพ็คเก็ตหลายถัง (Video Relay Service Recording System Using Packet-Capture Multiple Buckets Architecture) ซึ่งมีความสามารถในการจัดเก็บและบันทึกบริการถ่ายทอดการสื่อสารแบบ Video Relay Service ได้อย่างอัตโนมัติ (Fully Automated) และมีความฉลาด (Intelligence) ในการวิเคราะห์ขั้นสูงเกี่ยวกับรูปแบบต่างๆ ของบริการสนทนา นอกจากนี้ได้นำเสนอการออกแบบสถาปัตยกรรมการประมวลผล PCAP (Packet Capture) แบบใหม่เรียกว่า Multiple Buckets Architecture ที่สามารถจัดการและประมวล Real-Time Interactive Multi Flows ได้อย่างมีประสิทธิภาพ และสามารถลดโอกาสเกิด Buffer Overflow ได้อย่างมาก เมื่อเปรียบเทียบกับการประมวลผลแบบทั่วไปซึ่งเป็น Single Bucket Architecture

เอกสารอ้างอิง

[1] S. McCanne and V. Jasobson, “The BSD Packet Filter: A New Architecture for User-level Packet Capture,” in Proc. USENIX Winter 1993 Technical Conference, San Diego, CA, USA, Jan. 25–29, 1993, pp. 259–269.

[2] *TCPDUMP & LiBPCAP*, The Tcpdump Group, 2010. [Online]. Available: https://www.tcpdump.org

[3] N. Bonelli, F. D. Vigna, S. Giordano and G. Procissi, “Packet Fan-Out Extension for the pcap Library,” *IEEE Transactions on Network and Service Management*, vol. 15, no. 3, 2018, pp. 976–990, doi: 10.1109/TNSM.2018.2828939.

[4] J. F. Zazo, M. Forconesi, S. Lopez-Buedo, G. Sutter and J. Aracil, “TNT10G: A High - Accuracy 10 GbE Traffic Player and Recorder for Multi-Terabyte Traces,” in *International Conference on ReConFigurable Computing and FPGAs*, Cancun, Mexico, Dec. 8–10, 2014, pp. 1-6, doi: 10.1109/ReConFig.2014.7032561.

[5] V. Duarte and N. Farruca, “Using libPcap for Monitoring Distributed Applications,” in *International Conference on High Performance Computing & Simulation*, Caen, France, Jun. 28–2, 2010, pp. 92–97, doi: 10.1109/HPCS.2010.5547144.

[6] J. Liu, F. Wang, S. Zhao, X. Wang, and S. Chen, “iMonitor, An APP-Level Traffic Monitoring and Labeling System for iOS Devices,” in *IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, New York, NY, USA, Aug. 1–3, 2019, pp. 211–218, doi: 10.1109/CSE/EUC.2019.00048.

[7] *WinPcap*, Riverbed Technology, 2018. [Online]. Available: https://www.winpcap.org

[8] *Win10Pcap*, Daiyuu Nobori, Computer Science, University of Tsukuba, 2022. [Online]. Available: https://www.win10pcap.org

[9] *Npcap: Windows Packet Capture Library & Driver*, Insecure.Com, 2021. [Online]. Available: https://nmap.org/npcap

[10] P. Orosz and T. Skopko, “Software-based Packet Capturing with High Precision Timestamping for Linux,” in *International Conference on Systems and Networks Communications*, Nice, France, Aug. 22–27, 2010, pp. 381–386, doi: 10.1109/ICSNC.2010.65.

[11] *SIP: Session Initiation Protocol*, RFC3261, J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley and E. Schooler, Jun. 2002. [Online]. Available: https://www.ietf.org/rfc/rfc3261.txt

[12] *SDP: Session Description Protocol*, RFC4566, M. Handley, V. Jacobson and C. Perkins, Jul. 2006. [Online]. Available: https://www.ietf.org/rfc/rfc4566.txt

[13] *RTP: A Transport Protocol for Real-Time Applications*, RFC3550, H. Schulzrinne, S. Casner, R.

Frederick and V. Jacobson, Jul. 2003. [Online]. Available: https://www.ietf.org/rfc/rfc3550.txt

[14] *RTP Payload Format for H.264 Video*, RFC6184, Y.-K. Wang, R. Even, T. Kristensen and R. Jesup, May. 2011. [Online]. Available: https://www.ietf.org/rfc/rfc6184.txt

[15] *RTP Payload Format for Scalable Video Coding*, RFC6190, S. Wenger, Y.-K. Wang, T. Schierl and A. Eleftheriadis, May. 2011. [Online]. Available: https://www.ietf.org/rfc/rfc6190.txt

[16] *RTP Payload Format for G.711*, RFC7655, M. Ramalho, Ed. P. Jones, N. Harada, M. Perumal and L. Miao, Nov. 2015. [Online]. Available: https://www.ietf.org/rfc/rfc7655.txt

[17] *RTP Profile for Audio and Video Conferences with Minimal Control*, RFC3551, H. Schulzrinne and S. Casner, Jul. 2003. [Online]. Available: https://www.ietf.org/rfc/rfc3551.txt

[18] *Real-time Transport Protocol (RTP) Payload for Comfort Noise (CN)*, RFC3389, R. Zopf, Sep. 2002. [Online]. Available: https://www.ietf.org/rfc/rfc3389.txt

[19] *Telepoint* "FFmpeg." ffmpeg.org https://www.ffmpeg.org (accessed Jul. 18, 2023).