

Design and Evaluation of a High-Availability Secure Mail Relay with OAuth2 Authentication for Legacy Systems

Pichit Thongdom^{1,*} and Tanun Jaruvitayakovit²

^{1,*}Department of Computer Engineering, College of Engineering and Technology, Dhurakij Pundit University,
Bangkok, 10210, Thailand; Email: 66130409@dpu.ac.th

²Department of Computer Engineering, College of Engineering and Technology, Dhurakij Pundit University,
Bangkok, 10210, Thailand; Email: tanun.jar@dpu.ac.th

*Corresponding author: Pichit Thongdom (66130409@dpu.ac.th)

Received: 2 November 2025; **Revised:** 15 January 2026;

Accepted: 19 January 2026; **Published:** 21 April 2026

Abstract

In an era of escalating cyber threats, the enforcement of Modern Authentication (OAuth 2.0) by Microsoft 365 has created a critical compatibility gap for legacy applications that cannot be modified, posing significant risks to data security and business continuity, particularly in email communication systems. This research proposes High-Availability Secure Mail Relay architecture. The core novelty lies in the development of an “OAuth2 Agent Module” an intermediary mechanism designed to seamlessly translate legacy authentication protocols into OAuth 2.0 standards without requiring source code modifications. This is integrated with a Service-Level High Availability (HA) system capable of detecting internal service failures. Experimental results confirm that (1) The proposed architecture achieves a 100% success rate in relaying emails using OAuth 2.0 standards (2) While the enhanced security processing introduces a slight latency increase to 2.158 seconds compared to traditional Basic Authentication 1.908 seconds, performance remains within acceptable limits and (3) The HA mechanism successfully executes service failover in an average of 1.312 seconds. This study contributes a practical framework for retrofitting legacy systems to comply with modern security mandates while ensuring resilience.

Keywords: Secure Mail Relay, Legacy Applications, Modern Authentication, OAuth2 Agent, Service-Level High Availability



Copyright © 2026 Journal of Computer and Creative Technology. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC-BY-NC-ND).

Citation:

Thongdom, P., & Jaruvitayakovit, T. (2026). Design and Evaluation of a High-Availability Secure Mail Relay with OAuth2 Authentication for Legacy Systems. *Journal of Computer and Creative Technology*, 4(1), e2971.
<https://doi.org/10.65205/jcct.2026.e2971>

การออกแบบและประเมินผลระบบส่งต่ออีเมลปลอดภัยที่มีความพร้อมใช้งานสูง ด้วยการยืนยันตัวตน OAuth2 สำหรับระบบดั้งเดิม

พิชิต ทองดำ^{1,*} และ ธนัญ จารุวิทย์โกวิท²

^{1,*}สาขาวิศวกรรมคอมพิวเตอร์ วิทยาลัยนวัตกรรมการด้านเทคโนโลยีและวิศวกรรมศาสตร์ มหาวิทยาลัยธุรกิจบัณฑิตย์
จังหวัดกรุงเทพมหานคร 10210 ประเทศไทย; Email: 66130409@dpu.ac.th

²สาขาวิชาวิศวกรรมคอมพิวเตอร์ วิทยาลัยนวัตกรรมการด้านเทคโนโลยีและวิศวกรรมศาสตร์ มหาวิทยาลัยธุรกิจบัณฑิตย์
จังหวัดกรุงเทพมหานคร 10210 ประเทศไทย; Email: tanun.jar@dpu.ac.th

*ผู้ประพันธ์บทความ: พิชิต ทองดำ (66130409@dpu.ac.th)

รับบทความ: 2 พฤศจิกายน 2568; แก้ไขบทความ: 15 มกราคม 2569;
ตอบรับบทความ: 19 มกราคม 2569; เผยแพร่บทความ: 21 เมษายน 2569

บทคัดย่อ

ท่ามกลางภัยคุกคามทางไซเบอร์ที่ทวีความรุนแรง มาตรฐานการยืนยันตัวตนสมัยใหม่บนโปรโตคอล OAuth 2.0 ของ Microsoft 365 ได้สร้างปัญหาความเข้ากันไม่ได้สำหรับแอปพลิเคชันดั้งเดิมที่ไม่สามารถแก้ไขซอร์สโค้ดได้ ส่งผลให้องค์กรเผชิญความเสี่ยงทั้งด้านความปลอดภัยและความต่อเนื่องทางธุรกิจ โดยเฉพาะในระบบสื่อสารผ่านอีเมล งานวิจัยนี้นำเสนอสถาปัตยกรรมระบบส่งต่ออีเมลปลอดภัยที่มีความพร้อมใช้งานสูง จุดเด่นอยู่ที่การพัฒนา “OAuth2 Agent Module” ซึ่งทำหน้าที่เป็นตัวกลางในการแปลงโปรโตคอลการยืนยันตัวตนจากระบบเดิมไปสู่มาตรฐาน OAuth 2.0 ได้อย่างราบรื่นโดยไม่ต้องแก้ไขแอปพลิเคชันดั้งเดิม และบูรณาการร่วมกับระบบความพร้อมใช้งานสูงที่สามารถตรวจสอบความผิดปกติได้ในระดับบริการ ผลการทดสอบเชิงประจักษ์ยืนยันว่า 1) สถาปัตยกรรมที่นำเสนอสามารถส่งต่ออีเมลด้วยมาตรฐาน OAuth 2.0 ได้สำเร็จ 100% 2) แม้กระบวนการความปลอดภัยที่เพิ่มขึ้นจะทำให้ความหน่วงเฉลี่ยเพิ่มขึ้นเล็กน้อยเป็น 2.158 วินาที เมื่อเทียบกับระบบการยืนยันตัวตนพื้นฐานแบบดั้งเดิม 1.908 วินาที แต่ยังคงอยู่ในเกณฑ์ที่ยอมรับได้ และ 3) ระบบ HA สามารถกู้คืนการทำงานเมื่อบริการล้มเหลวได้ภายในเวลาเฉลี่ย 1.312 วินาที งานวิจัยนี้จึงเป็นแนวทางสำคัญในการยกระดับความปลอดภัยให้ระบบดั้งเดิมสามารถทำงานร่วมกับมาตรฐานสมัยใหม่ได้อย่างมั่นคงและปลอดภัย

คำสำคัญ: ระบบส่งต่ออีเมลปลอดภัย, แอปพลิเคชันดั้งเดิม, การยืนยันตัวตนสมัยใหม่, โอออธ 2 เอเจนต์, ความพร้อมใช้งานสูงระดับบริการ



Copyright © 2026 Journal of Computer and Creative Technology. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC-BY-NC-ND).

การอ้างอิง:

พิชิต ทองดำ, และ ธนัญ จารุวิทย์โกวิท. (2569). การออกแบบและประเมินผลระบบส่งต่ออีเมลปลอดภัยที่มีความพร้อมใช้งานสูงด้วยการยืนยันตัวตน OAuth2 สำหรับระบบดั้งเดิม. วารสารคอมพิวเตอร์และเทคโนโลยีสร้างสรรค์, 4(1), e2971.

<https://doi.org/10.65205/jcct.2026.e2971>

1. บทนำ (Introduction)

ปัจจุบันอีเมลยังคงเป็นเครื่องมือสื่อสารที่สำคัญที่สุดสำหรับกระบวนการทางธุรกิจในระดับองค์กร อย่างไรก็ตามท่ามกลางภัยคุกคามทางไซเบอร์ที่ทวีความรุนแรง ผู้ให้บริการอีเมลอย่าง Microsoft 365 ได้ยกระดับมาตรฐานความปลอดภัยโดยบังคับใช้การยืนยันตัวตนสมัยใหม่ (Modern Authentication) บนมาตรฐาน OAuth 2.0 และประกาศยุติการสนับสนุนการยืนยันตัวตนพื้นฐาน (Basic Authentication) สำหรับการส่งอีเมลภายในเดือนเมษายน ปี 2026 (The Exchange Team, 2024) การเปลี่ยนแปลงนี้นับเป็นวิกฤตสำคัญสำหรับองค์กรที่มีระบบแอปพลิเคชันดั้งเดิมจำนวนมาก ซึ่งถูกจำกัดให้รองรับเพียงการยืนยันตัวตนพื้นฐานด้วยรหัสผ่านตายตัว (Static Password) และไม่สามารถปรับปรุงซอร์สโค้ดให้รองรับมาตรฐานใหม่ที่ใช้แอคเซสโทเคน (Access Token) ได้ เนื่องจากความซับซ้อนทางเทคนิค ค่าใช้จ่ายที่สูงและความเสี่ยงต่อเสถียรภาพของระบบ (Chatterjee et al., 2024) หากไม่มีแนวทางแก้ไขที่เหมาะสม องค์กรจะเผชิญกับความเสียหายที่ระบบงานสำคัญต้องหยุดชะงักทางธุรกิจ (Business Disruption) พื้นที่ที่มาตรการบังคับใช้มีผล เนื่องจากแอปพลิเคชันดั้งเดิมไม่สามารถเชื่อมต่อเพื่อส่งอีเมลทำธุรกรรมได้อีกต่อไป นอกจากนี้ในฐานะที่ระบบส่งต่ออีเมลทำหน้าที่เป็นตัวกลาง (Central Gateway) ขององค์กร ความพร้อมใช้งานสูง (High Availability) จึงเป็นปัจจัยที่ต้องพิจารณาควบคู่กันเพื่อป้องกันความล้มเหลวเพียงจุดเดียว (Single Point of Failure) ที่อาจส่งผลให้ระบบงานทั้งหมดหยุดชะงัก จากการทบทวนวรรณกรรมและเทคโนโลยีที่เกี่ยวข้อง พบว่าแนวทางแก้ไขปัญหานี้ในปัจจุบันยังมีช่องว่างของงานวิจัย (Research Gap) ที่สำคัญ ดังนี้ 1) ข้อจำกัดของระบบส่งต่ออีเมลที่มีอยู่ ซอฟต์แวร์ตัวแทนส่งอีเมล (Mail Transfer Agent: MTA) ปัจจุบัน เช่น Postfix หรือ IIS 6.0 SMTP ยังไม่รองรับการแปลงโปรโตคอลจาก Basic Auth เป็น OAuth 2.0 สำหรับแอปพลิเคชันดั้งเดิม 2) ขาดการบูรณาการระบบความพร้อมใช้งานสูง งานวิจัยส่วนใหญ่มักแยกส่วนระหว่างเรื่องความปลอดภัยและความพร้อมใช้ ยังไม่พบงานวิจัยใดที่นำเสนอสถาปัตยกรรมที่บูรณาการ OAuth2 Agent เข้ากับ ตรวจสอบความผิดปกติระดับบริการ (Service-Level) ในระบบเดียวกัน และ 3) ระบบตรวจสอบที่ไม่ครอบคลุม ไซลูชันความพร้อมใช้งานสูงทั่วไปมักตรวจสอบเพียงระดับโหนด (Node Level) แต่ไม่สามารถตรวจสอบความผิดปกติระดับบริการ เช่น กรณีที่บริการหยุดทำงานขณะเครื่องแม่ข่ายยังออนไลน์อยู่ ซึ่งก่อให้เกิดภาวะการบริการล้มเหลวที่ระบบไม่สามารถตรวจจับได้ (Silent Failure)

ดังนั้นเพื่อปิดช่องว่างดังกล่าว งานวิจัยนี้นำเสนอสถาปัตยกรรมระบบส่งต่ออีเมลปลอดภัย (Secure Mail Relay) ที่ออกแบบมาเพื่อเชื่อมช่องว่างระหว่างระบบดั้งเดิมและมาตรฐานความปลอดภัยใหม่ โดยมีคุณค่าทางวิชาการที่สำคัญ (Main Contributions) ดังนี้

1. การนำเสนอนวัตกรรม OAuth2 Agent สำหรับการยืนยันตัวตน โดยได้พัฒนาโมดูล “OAuth2 Agent” ทำหน้าที่เป็นตัวกลางในการแปลงการยืนยันตัวตนจากระบบดั้งเดิม (Legacy Application) ไปสู่มาตรฐาน OAuth 2.0 ได้อย่างราบรื่น ช่วยให้แอปพลิเคชันสามารถใช้งานต่อไปได้โดยไม่ต้องแก้ไขซอร์สโค้ด
2. นำเสนอกลไกการตรวจสอบสุขภาพระบบ (Custom Health Check) ร่วมกับ Keepalived เพื่อตรวจจับความผิดปกติเชิงลึกในระดับแอปพลิเคชันของ Postfix และ Dovecot พร้อมสามารถกู้คืนระบบได้อัตโนมัติ (Failover)
3. การประเมินผลเชิงประจักษ์ (Experimental Evaluation) ดำเนินการทดสอบและเปรียบเทียบประสิทธิภาพทั้งด้านความปลอดภัย (Security), ความหน่วง (Latency) และระยะเวลาในการกู้คืนระบบ (Failover Time) เพื่อยืนยันประสิทธิภาพและความเป็นไปได้ในการนำไปใช้งานจริงในระดับองค์กร

2. วัตถุประสงค์งานวิจัย (Research Objectives)

1. เพื่อออกแบบสถาปัตยกรรม Secure Mail Relay ที่รองรับ OAuth 2.0 Authentication สำหรับ Legacy Application
2. เพื่อประเมินประสิทธิภาพด้านความปลอดภัยและเวลาในการส่ง (latency) เมื่อเทียบกับ Basic Authentication และระบบ Relay แบบอื่น
3. เพื่อประเมินความสามารถของระบบ High Availability ในระดับ Service-Level ผ่าน Keepalived และตรวจสอบความถูกต้องของระบบแจ้งเตือน

3. การทบทวนวรรณกรรมและทฤษฎีที่เกี่ยวข้อง (Literature Review)

3.1 ทฤษฎีและเทคโนโลยีพื้นฐาน (Theoretical Background)

งานวิจัยนี้ประยุกต์ใช้ทฤษฎีพื้นฐานสำคัญ 2 ประการ เพื่อออกแบบสถาปัตยกรรมที่ปลอดภัยและเข้ากันได้กับมาตรฐานสมัยใหม่ ดังนี้

1. วิวัฒนาการความปลอดภัยของ SMTP และข้อจำกัด (SMTP Security Evolution and Limitations) โพรโทคอล SMTP ดั้งเดิม RFC 5321 (Klensin, 2008) ถูกออกแบบโดยขาดกลไกความปลอดภัยพื้นฐาน ทำให้ข้อมูลถูกส่งเป็น Plaintext แม้ภายหลังจะมีมาตรฐาน STARTTLS RFC 3207 (Hoffman, 2002) เข้ามาช่วยยกระดับการเข้ารหัสช่องทางสื่อสาร (Encryption) ในระหว่างการรับส่งข้อมูล (Transit) แต่กลไกดังกล่าวยังไม่ครอบคลุมถึงการพิสูจน์ตัวตนที่แข็งแกร่ง โดยยังคงพึ่งพาการยืนยันตัวตนพื้นฐานที่ใช้ User และ Password ซึ่งมีความเสี่ยงสูงต่อการถูกโจมตีแบบ Brute Force และ Password Spraying นำมาสู่การบังคับใช้มาตรฐานใหม่ที่ยกเลิกการสนับสนุนการยืนยันตัวตนพื้นฐานในระบบอีเมลองค์กร

2. กรอบการทำงาน OAuth 2.0 สำหรับแอปพลิเคชัน (OAuth 2.0 for Non-Interactive Applications) เพื่อแก้ปัญหาความปลอดภัยของรหัสผ่าน มาตรฐาน OAuth 2.0 RFC 6749 (Hardt, 2012) ได้ถูกนำมาใช้แทนที่ โดยงานวิจัยนี้มุ่งเน้นที่กระบวนการ Client Credentials Grant Flow (Microsoft, 2025a) ซึ่งออกแบบมาสำหรับการสื่อสารแบบเครื่องต่อเครื่อง (Machine-to-Machine: M2M) โดยเฉพาะ ทฤษฎีนี้อนุญาตให้แอปพลิเคชัน ในที่นี้คือ OAuth2 Agent สามารถยืนยันตัวตนด้วย Client Certificate หรือ Client Secret เพื่อขอแอคเซสโทเคน จาก Authorization Server ได้โดยตรงโดยไม่ต้องอาศัยปฏิสัมพันธ์กับผู้ใช้งาน (User Interaction) จากนั้นจึงนำ Token ไปยืนยันตัวตนผ่านกลไก SASL XOAUTH2 ซึ่งเป็นหัวใจสำคัญที่ทำให้ระบบส่งต่ออีเมลปลอดภัย สามารถทำงานร่วมกับ Microsoft 365 ได้อย่างปลอดภัย

3.2 เทคโนโลยีและเครื่องมือที่เกี่ยวข้อง (Related Technologies and Tools)

งานวิจัยนี้เลือกใช้เทคโนโลยีแบบ Open Source โดยจัดกลุ่มตามหน้าที่การทำงานในสถาปัตยกรรมระบบ ดังนี้

1. แกนหลักของระบบส่งต่ออีเมล (Core Relay Engine) Postfix ถูกใช้เป็น MTA ซึ่งมีจุดเด่นที่สถาปัตยกรรมแบบหลายโปรเซส (Multi-process Architecture) ที่เน้นการแยกสิทธิ์การทำงาน (Privilege Separation) (Postfix, n.d.) Postfix ถูกตั้งค่าเป็นระบบส่งต่ออีเมลปลอดภัยโดยรับอีเมลจากแอปพลิเคชันดั้งเดิม ทำงานร่วมกับ Dovecot ที่ทำหน้าที่เป็น SASL สำหรับ Postfix เนื่องจาก Postfix ไม่ได้ทำหน้าที่พิสูจน์ตัวตนกับแอปพลิเคชันดั้งเดิม (Dovecot Team, n.d.) โดย Postfix จะส่งต่อคำร้องขอ SMTP AUTH ไปให้ Dovecot ตรวจสอบผ่าน UNIX Socket

2. กรอบการพัฒนา OAuth2 Agent (Agent Development Framework) aiosmtpd ถูกเลือกใช้เป็นไลบรารีหลักในการสร้าง OAuth2 Agent ด้วยสถาปัตยกรรมแบบ Asynchronous (The aiosmtpd developers, 2025) ซึ่งสามารถสร้าง Custom Handler เพื่อฝั่งตรรกะการขอตามมาตรฐาน OAuth 2.0 และการส่งต่ออีเมลเข้ากับกระบวนการ SMTP ได้โดยตรง ในส่วนของการจัดการสิทธิ์ งานวิจัยนี้ได้ใช้ไลบรารี msal (Microsoft, 2025b) เพื่อสื่อสารกับ Microsoft Entra ID ในฐานะผู้ให้บริการยืนยันตัวตน (Identity Provider: IdP) สำหรับการขอรับแอคเซสโทเคน ผ่านกระบวนการ Client Credentials Flow และใช้โมดูลมาตรฐาน smtplib (Python Software Foundation, 2025) ในการสร้างการเชื่อมต่อ STARTTLS และส่งแอคเซสโทเคนที่ได้รับผ่านกลไก SASL โดยใช้ AUTH XOAUTH2 ไปยัง Microsoft 365 รวมถึงใช้พัฒนาโปรแกรมจำลองการส่งอีเมลเพื่อการทดสอบ นอกจากนี้ งานวิจัยนี้ได้ใช้ไลบรารี Cryptography (Cryptography, 2023) เพื่อยกระดับความปลอดภัยด้วยสถาปัตยกรรม Fernet ซึ่งเป็นการเข้ารหัสแบบสมมาตร (Symmetric Encryption) ที่ผสานอัลกอริทึม AES-128 ร่วมกับกลไก HMAC-SHA256 เพื่อรักษาความถูกต้องของข้อมูล (Integrity) และป้องกันการลักลอบแก้ไขไฟล์โทเคน

3. กลไกความพร้อมใช้งานสูง (High Availability Mechanism) Keepalived ทำหน้าที่บริหารจัดการ (Virtual IP: VIP) ผ่านโปรโตคอล VRRP โดยจุดเด่นที่งานวิจัยนี้นำมาใช้คือความสามารถในการเรียกใช้ Custom Health Check Script (Cassen & Contributors, 2017) ทำให้สามารถขยายขอบเขตการตรวจสอบจากระดับโหนดไปสู่ระดับบริการได้

3.3 งานวิจัยที่เกี่ยวข้อง

จากการทบทวนวรรณกรรม โดยการสังเคราะห์องค์ความรู้จากงานวิจัยที่เกี่ยวข้องและจำแนกออกเป็น 3 กลุ่มหลัก เพื่อระบุช่องว่างทางวิชาการและแนวทางประยุกต์ใช้ ดังนี้

1. สถาปัตยกรรมระบบอีเมลและความปลอดภัย งานวิจัยส่วนใหญ่ยอมรับประสิทธิภาพของ Postfix ในฐานะ MTA ที่มีความเสถียรสูงโดย Sharabov et al. (2024) และ Oliveira et al. (2022) ได้นำเสนอการปรับแต่ง Postfix เพื่อการกรองสแปมและบัญชีดำแบบเรียลไทม์ ซึ่งยืนยันถึงความยืดหยุ่นของ Postfix ในการรองรับส่วนขยาย (Extensions) อย่างไรก็ตาม งานวิจัยกลุ่มนี้ยังมุ่งเน้นเพียงความปลอดภัยในการกรองเนื้อหา และประสิทธิภาพของ Postfix แต่ยังไม่ครอบคลุมถึงแนวทางแก้ปัญหาการยืนยันตัวตนสมัยใหม่ สำหรับระบบแอปพลิเคชันดั้งเดิม ซึ่งเป็นโจทย์หลักของงานวิจัยนี้

2. สถาปัตยกรรมตัวกลางและความปลอดภัย ในบริบทของระบบสมัยใหม่ Gbenle et al. (2022) ได้นำเสนอแนวทางการใช้ API Gateway เป็นตัวกลางในการรักษาความปลอดภัย (Secure Gateway) สำหรับระบบแบบกระจายศูนย์ (Distributed Systems) โดยทำหน้าที่จัดการกระบวนการยืนยันตัวตนด้วย OAuth 2.0 และ JSON Web Tokens (JWT) แทนแอปพลิเคชันหลังบ้าน ซึ่งช่วยแยกกระบวนการยืนยันตัวตนออกจากตรรกะของแอปพลิเคชัน (Decoupling) และลดความเสี่ยงจากการเปิดเผยข้อมูลสำคัญ อย่างไรก็ตาม แม้แนวคิดการใช้ตัวกลาง จะได้รับการพิสูจน์แล้วว่ามีประสิทธิภาพในการจัดการโทเคนและการรักษาความปลอดภัยสำหรับสถาปัตยกรรม แต่ปัจจุบันยังขาดงานวิจัยที่นำสถาปัตยกรรมดังกล่าวมาประยุกต์ใช้กับโปรโตคอลอีเมลอย่าง SMTP เพื่อแก้ปัญหาความเข้ากันไม่ได้ของระบบแอปพลิเคชันดั้งเดิมที่ไม่รองรับการยืนยันตัวตนสมัยใหม่ ซึ่งเป็นช่องว่างที่งานวิจัยนี้มุ่งเน้นแก้ไข

3. ความพร้อมใช้งานสูงและการตรวจสอบระบบ ในด้านความพร้อมใช้งาน Putranto (2019) ได้เสนอการใช้ Keepalived และ VRRP เพื่อทำกลไกการสลับทำงานอัตโนมัติ ในระดับโหนด แต่แนวทางดังกล่าวยังมีข้อจำกัดที่ไม่สามารถตรวจจับความผิดปกติภายในระดับบริการได้ เช่น กรณีบริการหยุดทำงานแต่เครื่องยังออนไลน์อยู่ เพื่อปิดช่องโหว่นี้จึงได้นำแนวคิด Service-Behavioral Monitoring ของ Faseeha et al. (2025) ที่เน้นการตรวจสอบพฤติกรรมของบริการ (Service Behavior) มาประยุกต์ใช้ในการพัฒนา Custom Health Check Script เพื่อให้ระบบความพร้อมใช้งานสูงมีความแม่นยำยิ่งขึ้น

4. การปรับปรุงระบบดั้งเดิมและความปลอดภัยของโทเคน สำหรับการจัดการระบบ Legacy นั้น Mishra (2020) เสนอกลยุทธ์ Encapsulation หรือการสร้าง Wrapper ห่อหุ้มระบบเดิมเพื่อเชื่อมต่อกับมาตรฐานใหม่ ซึ่งเป็นรากฐานแนวคิดของการพัฒนา "OAuth2 Agent" ในงานวิจัยนี้ นอกจากนี้ เพื่อความปลอดภัยในการจัดการโทเคน Matcha & Kumar (2025) ระบุว่า การเข้ารหัสและจำกัดสิทธิ์การเข้าถึงไฟล์โทเคนสามารถลดความเสี่ยงจากการโจรกรรมได้ถึงร้อยละ 88.6 ซึ่งงานวิจัยนี้ได้นำหลักการนี้มาใช้ในการออกแบบระบบแคชโทเคน (Token Cache) ให้มีความปลอดภัยสูง

4. กรอบแนวคิดงานวิจัย (Conceptual Framework)

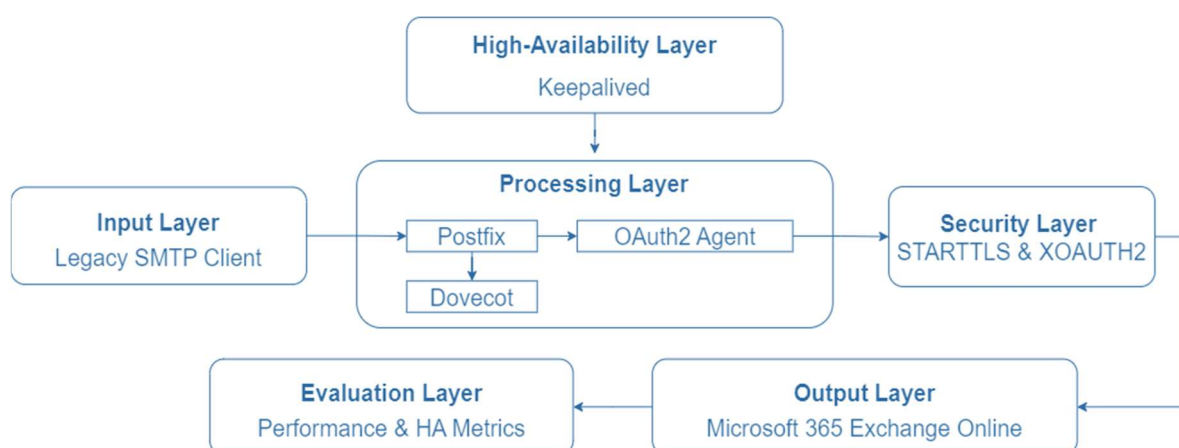


Figure 1. Conceptual Framework for Design and Evaluation of a High-Availability Secure Mail Relay with OAuth2 Authentication for Legacy Systems

แผนภาพกรอบแนวคิด Figure 1. แสดงสถาปัตยกรรมระบบในรูปแบบ System-Based Conceptual Framework โดยแบ่งองค์ประกอบออกเป็นชั้น (Layers) ตามหน้าที่การทำงานและการไหลของข้อมูล (Data Flow) ดังนี้

1. ชั้นนำเข้าข้อมูล (Input Layer) ประกอบด้วย Legacy SMTP Client ทำหน้าที่เป็นจุดเริ่มต้นของกระบวนการ โดยส่งคำขอส่งอีเมลผ่านโปรโตคอล SMTP (Plaintext) เข้ามายังระบบ
2. ชั้นประมวลผล (Processing Layer) ทำหน้าที่เป็นแกนหลักในการประมวลผล ประกอบด้วย Postfix เป็น MTA ด้านหน้าทำหน้าที่รับการเชื่อมต่อและบริหารจัดการคิว (Queue Management), Dovecot ทำหน้าที่ตรวจสอบสิทธิ์ในการยืนยันตัวตนของ Client ผ่านกลไก (Simple Authentication and Security Layer: SASL) และ OAuth2 Agent นวัตกรรมที่รับช่วงต่อจาก Postfix เพื่อแปลงการยืนยันตัวตนไปสู่มาตรฐานสมัยใหม่
3. ชั้นความปลอดภัย (Security Layer) เป็นชั้นที่ควบคุมมาตรฐานความปลอดภัยในการรับส่งข้อมูล ประกอบด้วย STARTTLS กลไกการเข้ารหัสช่องทางการสื่อสารให้ปลอดภัย (Encryption) และ XOAUTH2 กลไก SASL ที่ใช้สำหรับการยืนยันตัวตนสมัยใหม่ ซึ่ง OAuth2 Agent ใช้เจรจากับผู้ให้บริการโดยใช้แอคเซสโทเคน แทนการใช้รหัสผ่าน
4. ชั้นส่งออกข้อมูล (Output Layer) คือปลายทาง Microsoft 365 Exchange Online ซึ่งเป็นผู้ให้บริการอีเมลที่รับข้อมูลผ่านการยืนยันตัวตนและเข้ารหัสเรียบร้อยแล้วไปดำเนินการต่อ
5. ชั้นความพร้อมใช้งานสูง (High-Availability Layer) ทำหน้าที่กำกับดูแลเสถียรภาพของระบบ (System Reliability) ครอบคลุมการทำงานของ Processing Layer ทั้งหมด โดยใช้ Keepalived เป็นกลไกหลักในการตรวจสอบสถานะ (Health Check) ของบริการ Postfix และ Dovecot และควบคุมการสลับระบบ (Failover) อัตโนมัติเมื่อเกิดเหตุขัดข้อง
6. ชั้นการประเมินผล (Evaluation Layer) เป็นส่วนประเมินผลลัพธ์ของระบบทั้งด้านประสิทธิภาพ (Performance) และความพร้อมใช้งาน ผ่านตัวชี้วัดสำคัญ ได้แก่ ความหน่วง (Latency), อัตราความสำเร็จ (Success Rate) และระยะเวลาการกู้คืนระบบ (Failover Time) โดยเปรียบเทียบผลลัพธ์กับระบบส่งต่ออีเมลแบบดั้งเดิม ได้แก่ IIS 6.0 SMTP Virtual Server, Stunnel TLS Proxy และ Postfix (Basic Authentication) เพื่อยืนยันประสิทธิภาพและความเหมาะสมในการนำมาใช้งานทดแทน

5. วิธีดำเนินงานวิจัย (Research Methodology)

การวิจัยนี้มุ่งเน้นการพัฒนาและประเมินประสิทธิภาพของระบบส่งต่ออีเมลปลอดภัย เพื่อแก้ปัญหาความเข้ากันไม่ได้ของโปรโตคอลการยืนยันตัวตนในระบบดั้งเดิม รวมไปถึงความมั่นคงของบริการส่งต่ออีเมลปลอดภัย เพื่อให้การดำเนินการวิจัยเป็นไปอย่างมีมาตรฐานและตรวจสอบความถูกต้องได้ (Validity) โดยได้กำหนดกรอบการดำเนินงานออกเป็น 5 ระยะสำคัญ ดังนี้

5.1 วิธีการออกแบบระบบ (System Design Method)

5.1.1 สถาปัตยกรรมเครือข่ายและซอฟต์แวร์ (Network and Software Architecture)

ระบบถูกออกแบบในลักษณะ Secure Mail Relay Gateway (Figure 2.) ทำหน้าที่เป็นตัวกลางระหว่างแอปพลิเคชันดั้งเดิมและ Microsoft 365 โดยมีองค์ประกอบสำคัญ คือ

1. จุดเชื่อมต่อ (Entry Point) ให้บริการผ่าน VIP เพียงจุดเดียว
2. คลัสเตอร์เซิร์ฟเวอร์ (Relay Cluster) ทำงานแบบ Active/Standby จำนวน 2 โหนด โดยภายในแต่ละโหนดประกอบด้วยซอฟต์แวร์โอเพนซอร์สและนวัตกรรมที่พัฒนาขึ้นใหม่ ดัง Figure 3. ซึ่งมีรายละเอียด ดังนี้
 - 1) Incoming Handling ใช้ Postfix (2)smtpd ร่วมกับ Dovecot (3)dovecot SASL เพื่อรับอีเมลและยืนยันตัวตนพื้นฐาน จากแอปพลิเคชันดั้งเดิม (1)Legacy Application
 - 2) จาก (7)OAuth2 Agent นวัตกรรมที่พัฒนาด้วย Python แทรกอยู่ระหว่างกระบวนการส่งต่อ ทำหน้าที่แปลงการยืนยันตัวตนเป็น OAuth 2.0 (XOAUTH2) เพื่อส่งต่อไปยัง Microsoft 365
 - 3) Queue Management (5)qmgr ออกแบบกลไกการสื่อสารระหว่างโพรเซส ให้ Postfix จัดการคิวตาม SMTP Response Code ที่ได้รับจาก OAuth2 Agent (2xx=Success, 4xx=Retry, 5xx=Drop)

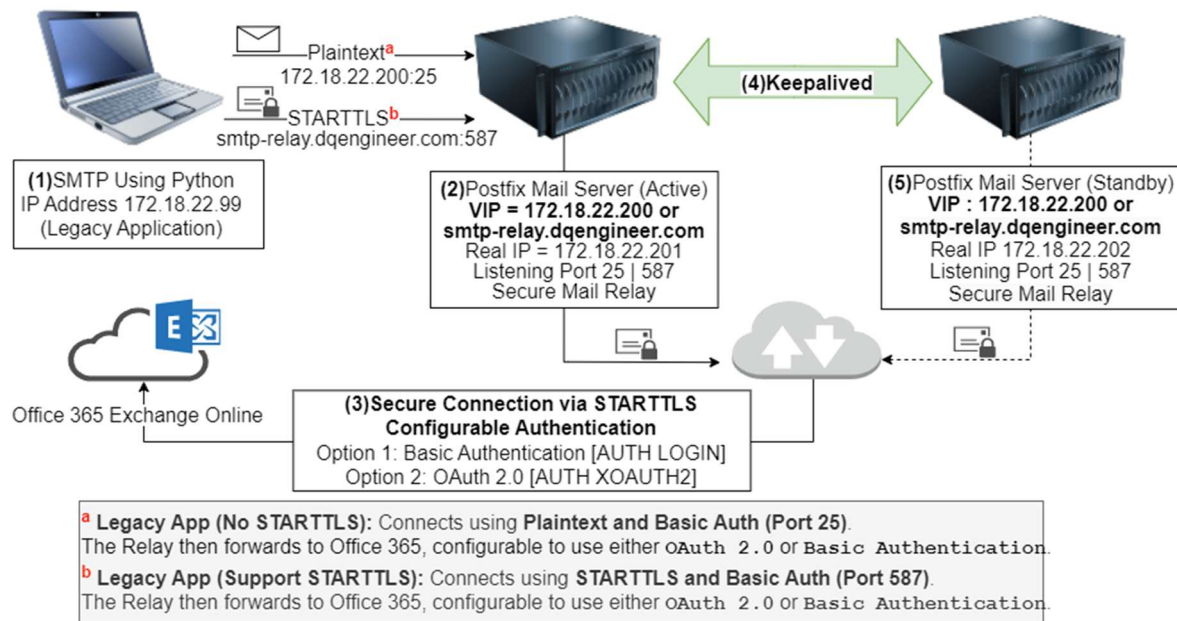


Figure 2. Architecture of a High Availability Secure Mail Relay for Legacy Applications

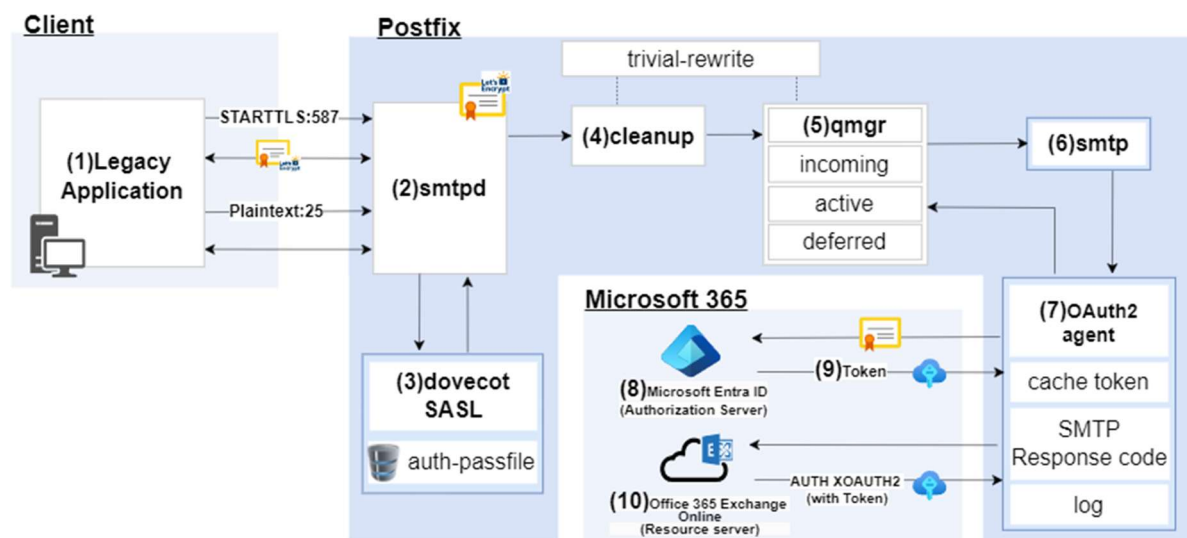


Figure 3. Architecture and Workflow of Postfix and Dovecot Integrating with the OAuth 2 Agent

5.1.2 โมเดลความปลอดภัย (Security Model)

งานวิจัยนี้กำหนดมาตรการป้องกันความเสี่ยงสำคัญ ดังนี้

1. Credential Theft Mitigation ยกเลิกการเก็บรหัสผ่านถาวร โดยใช้แอคเซสโทเคนที่มีอายุจำกัด (Time-limited) และการยืนยันตัวตนเพื่อขอแอคเซสโทเคนของ OAuth2 Agent กับ IdP โดยใช้ Self-signed Certificate (RSA 4096-bit) แทนการใช้ Client Secret

2. Token Management and Protection บริหารจัดการโทเคนที่มีอายุใช้งาน 1 ชั่วโมง โดยจัดเก็บในรูปแบบข้อมูลเข้ารหัส (Encrypted Binary) ร่วมกับกลไก HMAC-SHA256 บนพื้นที่จำกัดสิทธิ์ (Restricted Access) เพื่อรักษาความถูกต้องของข้อมูล (Integrity) พร้อมกลไกขอแอคเซสโทเคนใหม่ล่วงหน้า 300 วินาที (Safety Buffer) เพื่อความต่อเนื่องของบริการ

5.1.3 การออกแบบความพร้อมใช้งาน (High Availability Design) เพื่อป้องกันจุดล้มเหลวเพียงจุดเดียว

ระบบใช้โปรโตคอล VRRP ผ่านซอฟต์แวร์ Keepalived โดยมีกลไกตรวจสอบ 2 ระดับ ดังนี้

1. ระดับโหนด โดยกำหนดให้มีการส่ง VRRP Advertisement ทุกๆ 0.3 วินาที หากเครื่องสำรองไม่ได้รับสัญญาณดังกล่าวติดต่อกันครบ 3 ครั้ง โหนดสำรอง จะทำการยึดครอง (Takeover) VIP อัตโนมัติ
2. ระดับบริการ ใช้ชุดคำสั่งเฉพาะ (Custom Script) ตรวจสอบสถานะของบริการภายใน Postfix และ Dovecot ทุก 0.2 วินาที หากล้มเหลวติดต่อกัน 3 ครั้ง ระบบจะส่งลดค่าความสำคัญ (Priority) ของโหนดหลัก เพื่อบังคับให้ทำการกู้คืนความล้มเหลว ไปยังโหนดสำรองอัตโนมัติ

5.1.4 การออกแบบระบบแจ้งเตือน (Monitoring) เพื่อให้ผู้ดูแลระบบรับทราบสถานะการทำงานแบบเรียลไทม์

งานวิจัยนี้ได้บูรณาการระบบแจ้งเตือนโดยใช้ฟังก์ชัน notify_master ของ Keepalived เป็นตัวจุดชนวน (Trigger) เมื่อเกิดการสลับโหนด (Failover) ระบบจะเรียกใช้ชุดคำสั่งเพื่อรวบรวมข้อมูลบริบทสำคัญ Hostname, VIP, IP Address และ Timestamp จากนั้นจัดรูปแบบเป็น JSON ก่อนส่งผ่าน HTTP POST ไปยัง Microsoft Power Automate Webhook เพื่อแจ้งเตือนเข้าสู่กลุ่มสนทนาบน Microsoft Teams โดยอัตโนมัติ

5.2 การออกแบบการทดลอง (Experimental Design)

5.2.1 สภาพแวดล้อมการทดลอง (Experimental Environment)

การทดลองดำเนินการบนสภาพแวดล้อมเสมือน (Virtual Environment) โดยมีการควบคุมคุณสมบัติของฮาร์ดแวร์ให้เหมือนกันทุกประการ ดังแสดงใน Table 1.

Table 1. Experimental Environment Specifications

Component	Specifications
Host Hardware	AMD Ryzen™ 5 7540U, 16 GB RAM (6400MHz), SSD NVMe Gen4, Vention USB-C to Gigabit Ethernet Adapter
Virtual Machine	4 vCPU, 4GB vRAM, Network 1 Gbps (NAT)
Proposed System OS	Ubuntu Server 24.04.2 LTS
Proposed Software	Postfix 3.8.6, Keepalived 2.2.8, Python 3.12.3
Proposed Library	Cryptography 41.0.7, msal 1.26.0, aiosmtpd 1.4.4.post2
Baseline Systems	IIS 6.0 SMTP (Windows Server 2019), Stunnel 5.63 (Ubuntu 24.04.2 LTS)
Tools	Wireshark 4.4.5, Tshark 4.2.2

5.2.2 การควบคุมตัวแปร (Experimental Control) เพื่อให้ผลการทดลองมีความเที่ยงตรง

กำหนดมาตรการควบคุมตัวแปร คือ Network Conditions ทดสอบในระบบเครือข่ายจำลองแบบปิด (Isolated Virtual Network) เพื่อขจัดปัจจัยรบกวน (Local Traffic Noise) และใช้วิธีการทดสอบซ้ำ (Replication) จำนวน 30 ครั้ง เพื่อเฉลี่ยความผันผวนของอินเทอร์เน็ตขาออก

5.3 ตัวชี้วัดและการวิเคราะห์ข้อมูล (Metrics and Data Analysis)

5.3.1 ตัวชี้วัด (Measurement Metrics)

1. Transmission Latency วัดระยะเวลาทั้งหมดของเซสชันการสื่อสาร (Total Session Duration) โดยคำนวณจากผลต่างเวลา ตั้งแต่แอปพลิเคชันดั้งเดิม เริ่มส่งแพ็กเก็ต TCP SYN ไปยัง Secure Mail Relay และรอจนกว่ากระบวนการส่งต่อ SMTP ไปยัง Microsoft 365 เสร็จสิ้นจนกระทั่งระบบส่งต่ออีเมลปลอดภัย ส่งแพ็กเก็ต TCP FIN, ACK เพื่อยุติการเชื่อมต่อ
2. Success Rate (%) สัดส่วนจำนวนอีเมลที่ส่งถึงผู้รับปลายทางสำเร็จ (Delivered)
3. Failover Time วัดระยะเวลาการตอบสนองต่อความล้มเหลว นับตั้งแต่ระบบเริ่มเกิดความผิดปกติของระดับบริการ และระดับโหนด จนถึงเวลาที่โหนดสำรอง ส่งสัญญาณ Gratuitous ARP เพื่อประกาศการยึดครอง VIP เกณฑ์การ

ยอมรับ (Threshold) โดยกำหนดเกณฑ์ไว้ที่ไม่เกิน 1.800 วินาที ซึ่งคำนวณจากผลรวมของระยะเวลาการคอยสูงสุดทางทฤษฎี (Theoretical Maximum Delay)

4. การวิเคราะห์ความปลอดภัยระดับโปรโตคอล (Protocol Security Analysis) เป็นการวัดผลเชิงคุณภาพผ่านการตรวจสอบแพ็กเก็ต (Packet Inspection) ใน 2 ระยะสำคัญ ได้แก่ 1) ระยะยืนยันตัวตน (Authentication Phase) ตรวจสอบกลไก SASL ที่ระบบนำเสนอ (เช่น AUTH PLAIN, LOGIN, XOAUTH2) และ 2) ระยะเจรจาการเข้ารหัส (Handshake Phase) ตรวจสอบแพ็กเก็ต Server Hello จาก Microsoft 365 เพื่อยืนยันเวอร์ชัน TLS และ Cipher Suite ที่ตกลงใช้งานจริง (Negotiated Parameters)

5.4 วิธีการเก็บรวบรวมข้อมูล (Data Collection)

การเก็บรวบรวมข้อมูลดำเนินการผ่าน 3 ขั้นตอนการทดสอบหลัก โดยบูรณาการเครื่องมือวิเคราะห์เครือข่ายร่วมกับเครื่องมือตรวจสอบของผู้ให้บริการ เพื่อให้ได้ข้อมูลที่ครบถ้วนและยืนยันความถูกต้องได้ (Cross-Validation) ดังนี้

1. กระบวนการทดสอบ (Testing Procedure) ดำเนินการทดสอบจำนวน 30 ครั้งต่อชุดการทดลอง โดยใช้วิธีการส่งแบบลำดับ (Sequential Execution) รอให้การส่งครั้งก่อนหน้าเสร็จสมบูรณ์ก่อนเริ่มครั้งถัดไป เพื่อขจัดตัวแปรแทรกซ้อนด้านการรอคิว (Queuing Delay)

2. การบันทึกข้อมูลเชิงเทคนิค (Technical Recording) ใช้เครื่องมือ Wireshark บันทึกการจราจรเครือข่ายและวิเคราะห์พารามิเตอร์ความปลอดภัย

3. การยืนยันผลลัพธ์ (Result Verification) ใช้เครื่องมือ Message Trace บนระบบ Microsoft 365 Exchange Online ในการติดตามสถานะการนำส่ง (Delivered) เพื่อตรวจสอบยืนยันความถูกต้องว่าอีเมลที่เซิร์ฟเวอร์ตอบรับได้ถูกส่งถึงผู้รับปลายทางอย่างสมบูรณ์จริง

5.5 การวิเคราะห์ข้อมูล (Data Analysis Method)

งานวิจัยนี้นำข้อมูลดิบ (Raw Data) จากการทดสอบทั้ง 3 ชุด มาประมวลผลและวิเคราะห์ทางสถิติเพื่อสรุปผลประสิทธิภาพ โดยแบ่งเป็น 2 ส่วนหลัก ดังนี้

1. การวิเคราะห์เชิงพรรณนา (Descriptive Analysis) สำหรับข้อมูลเชิงปริมาณด้านเวลา (Transmission Latency และ Failover Time) โดยคำนวณค่าสถิติสำคัญ 7 ประการ เพื่ออธิบายพฤติกรรมของระบบอย่างละเอียด ได้แก่ 1) ค่าต่ำสุด (Min) และค่าสูงสุด (Max) เพื่อแสดงขอบเขตของข้อมูล (Data Range) 2) ค่าเฉลี่ย (Mean) เพื่อเป็นตัวแทนประสิทธิภาพโดยรวม 3) ส่วนเบี่ยงเบนมาตรฐาน (S.D.) เพื่อวัดความเสถียร (Stability) ของระบบ 4) ค่ามัธยฐาน (Median) เพื่อดูแนวโน้มเข้าสู่ส่วนกลางโดยลดผลกระทบจากค่าที่ผิดปกติ 5) ค่าเปอร์เซ็นต์ไทล์ที่ 95 (95th Percentile) เพื่อประเมินประสิทธิภาพในสถานการณ์ที่แย่ที่สุด และ 6) ค่าผิดปกติ (Outliers) ระบุโดยใช้เกณฑ์ Interquartile Range ($1.5 \times IQR$) เพื่อแยกแยะข้อมูลที่มีความล่าช้าผิดปกติอันเกิดจากความไม่เสถียรชั่วคราว

2. การวิเคราะห์เปรียบเทียบและนำเสนอผล (Comparative Analysis & Visualization)

1) เปรียบเทียบสมรรถนะระหว่างระบบที่นำเสนอกับระบบส่งต่ออีเมลที่มีอยู่ดั้งเดิมซึ่งรองรับเพียงการยืนยันตัวตนพื้นฐาน โดยนำเสนอผ่านแผนภาพกล่อง (Box Plot) เพื่อแสดงการกระจายตัวและเปรียบเทียบค่ากลางอย่างชัดเจน

2) เปรียบเทียบพารามิเตอร์ความปลอดภัยที่ได้จากการตรวจสอบแพ็กเก็ต (Packet Inspection) เทียบกับมาตรฐานความปลอดภัยสากล

6. ผลการวิจัย (Results)

ส่วนนี้จะนำเสนอผลการทดลองเชิงประจักษ์เพื่อตอบวัตถุประสงค์การวิจัยทั้ง 3 ข้อ โดยแบ่งการนำเสนอผลออกเป็น 3 ส่วน ได้แก่ 1) ผลการออกแบบสถาปัตยกรรม 2) ผลการประเมินประสิทธิภาพด้านความปลอดภัยและเวลาในการส่ง และ 3) ผลการประเมินความพร้อมใช้งานสูง

6.1 ผลการออกแบบสถาปัตยกรรม Secure Mail Relay

งานวิจัยนี้ได้สถาปัตยกรรมระบบส่งต่ออีเมลปลอดภัยที่สามารถทำงานร่วมกับแอปพลิเคชันดั้งเดิมได้สำเร็จ ดังแสดงใน Figure 2. โดยผลการพัฒนาส่วนประกอบซอฟต์แวร์ตาม Figure 3. พบว่าโมดูลที่พัฒนาขึ้น OAuth2 Agent สามารถ

บูรณาการร่วมกับ Postfix และ Dovecot ได้อย่างสมบูรณ์ ส่งผลให้ระบบมีคุณสมบัติทางเทคนิคครบถ้วนตามการออกแบบ
ดังนี้

1. สามารถรับการเชื่อมต่อแบบ Plaintext หรือ STARTTLS จากแอปพลิเคชันดั้งเดิมได้
2. สามารถแปลงการยืนยันตัวตนเป็น OAuth 2.0 (XOAUTH2) เพื่อส่งต่ออีเมลไปยัง Microsoft 365 ได้จริง
3. มีกลไกการทำงานความพร้อมใช้สูงในระดับบริการและระดับโหนดแบบ Active/Standby ผ่าน Keepalived

ที่พร้อมสำหรับการทดสอบประสิทธิภาพในลำดับถัดไป

6.2 ผลการประเมินประสิทธิภาพด้านความปลอดภัยและเวลาในการส่ง

การเปรียบเทียบประสิทธิภาพด้านความปลอดภัยและระยะเวลาในการส่ง ระหว่างระบบที่นำเสนอกับระบบดั้งเดิม
งานวิจัยนี้ได้ดำเนินการทดสอบและวิเคราะห์ผลลัพธ์ดังนี้

Table 2. Test Results of SMTP Commands and Email Delivery

Scenario	Success (%)	Auth (Pre-TLS)	Auth (Relay)	Auth (Office 365)	TLS Version	Cipher Suite
Legacy to Technologyland	100.00	PLAIN	-	-	-	-
Legacy to O365 (Direct)	0.00	-	-	-	-	-
Modern App to O365 (Direct)	100.00	-	-	LOGIN	1.3	TLS_AES_256_GCM_SHA384
Relay Postfix (Basic Auth)	100.00	-	PLAIN	LOGIN	1.3	TLS_AES_256_GCM_SHA384
Relay Stunnel (Basic Auth)	100.00	-	LOGIN	LOGIN	1.3	TLS_AES_256_GCM_SHA384
Relay IIS 6.0 (Basic Auth)	100.00	-	IP	LOGIN	1.2	TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
Relay OAuth2 Agent (Modern Auth)	100.00	-	PLAIN	XOAUTH2	1.3	TLS_AES_256_GCM_SHA384

Table 3. Descriptive Statistics of SMTP Transmission Latency (Time in Seconds) (n=30)

Solution	Min	Max	Mean	Median	S.D.	95th Percentile
Postfix	1.660	2.370	1.908	1.875	0.172	2.177
Postfix OAuth2 Agent	1.870	3.740	2.158	2.095	0.350	2.580
Stunnel TLS Proxy	2.030	2.420	2.148	2.120	0.102	2.352
IIS 6.0 SMTP Virtual Server	2.000	2.870	2.194	2.130	0.175	2.401

จาก Table 2. พบว่าระบบที่นำเสนอสามารถแก้ปัญหาความเข้ากันไม่ได้ของโปรโตคอลได้สมบูรณ์ โดยมีอัตรา
ความสำเร็จ ในการส่งอีเมลอยู่ที่ร้อยละ 100 และสามารถยกระดับความปลอดภัยจากการยืนยันตัวตนพื้นฐาน ไปสู่การ
ยืนยันตัวตนสมัยใหม่ (XOAUTH2) ได้สำเร็จ อีกทั้งยังมี TLS 1.3 และ Cipher Suite ที่มีความแข็งแกร่ง เหนือกว่า IIS 6.0
ที่รองรับเพียง TLS 1.2 ยืนยันว่าสถาปัตยกรรมที่นำเสนอสามารถยกระดับความปลอดภัยไปสู่มาตรฐาน OAuth 2.0 ได้สำเร็จ

จาก Table 3. และการกระจายตัวของข้อมูลใน Figure 4. พบว่า ระบบที่นำเสนอมีค่าความหน่วงเฉลี่ยอยู่ที่ ($\bar{x}$ = 2.158) ซึ่งสูงกว่าระบบ Postfix แบบดั้งเดิม ($\bar{x}$ = 1.908) และ Stunnel ($\bar{x}$ = 2.148) เพียงเล็กน้อย ส่วนต่างนี้สะท้อนถึงภาระงานส่วนเพิ่ม (Overhead) ของกระบวนการแอคเซสโทเคนสอดคล้องกับค่าผิดปกติ (Outlier) สูงสุดที่ 3.740 วินาที ที่เกิดจากรอบการทำงานของการขอแอคเซสโทเคนใหม่เมื่อหมดอายุ ซึ่งถือว่าอยู่ในเกณฑ์ที่ยอมรับได้แล้กับความปลอดภัยที่สูงขึ้น แต่ยังเร็วกว่าระบบดั้งเดิมอย่าง IIS 6.0 ($\bar{x}$ = 2.194) ยืนยันว่าประสิทธิภาพด้านระยะเวลาใกล้เคียงกับระบบส่งต่ออีเมลดั้งเดิม

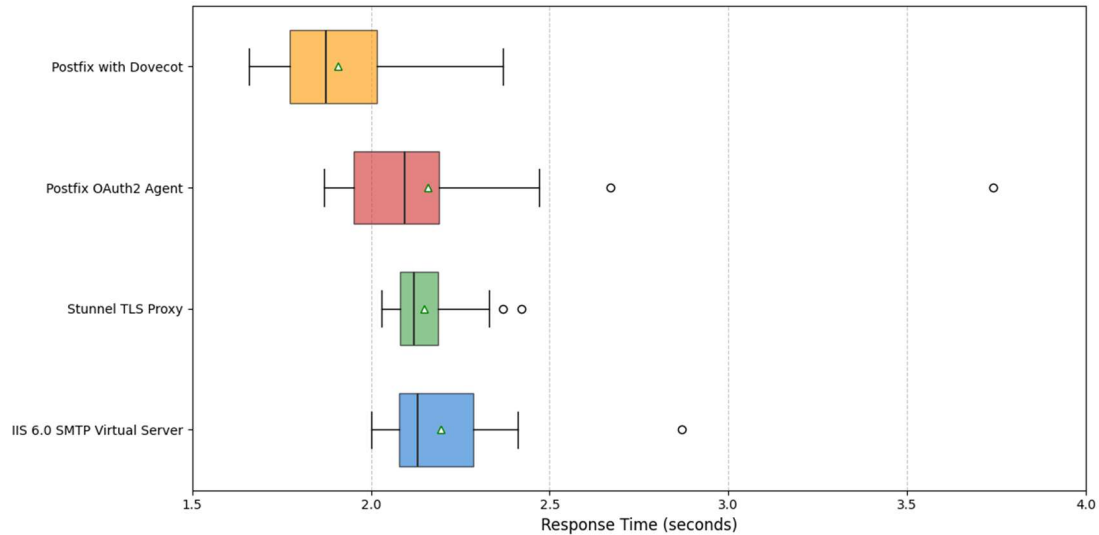


Figure 4. Comparison of SMTP Transmission Latency Distribution Across Different Secure Mail Relay Solutions

6.3 ผลการทดสอบความพร้อมใช้งานสูง

การประเมินความสามารถของระบบความพร้อมใช้งานสูง รวมไปถึงความแม่นยำของระบบแจ้งเตือน งานวิจัยนี้ได้จำลองสถานการณ์ความล้มเหลวและได้ผลลัพธ์ดังนี้

Table 4. High Availability Failover Performance Statistics (Time in Seconds) (n=10)

Failure Scenario	Min	Mean	Median	Max	S.D.	95th Percentile	Threshold	Result
Node Failure (Network)	0.783	0.863	0.860	0.997	0.071	0.962	< 1.800	Pass
Service Failure (Dovecot)	0.918	1.275	1.204	1.758	0.347	1.750	< 1.800	Pass
Service Failure (Postfix)	0.896	1.312	1.334	1.634	0.264	1.623	< 1.800	Pass

จาก Table 4. และ Figure 5. พบว่าระบบใช้เวลาในการกู้คืนกรณีโหนดล้มเหลวเฉลี่ย ($\bar{x}$ = 0.863) วินาที ในขณะที่กรณีที่บริการล้มเหลวของ Postfix และ Dovecot ใช้เวลาเฉลี่ย ($\bar{x}$ = 1.312) และ ($\bar{x}$ = 1.275) ตามลำดับ แม้การกู้คืนระดับบริการจะใช้เวลามากกว่าเล็กน้อยจากกระบวนการตรวจสอบของ Custom Script แต่ค่าที่วัดได้ทั้งหมดยังคงอยู่ภายใต้เกณฑ์การยอมรับที่กำหนด (< 1.800 วินาที) ผลลัพธ์นี้ยืนยันประสิทธิภาพของกลไกตรวจสอบระดับบริการ และความถูกต้องของระบบแจ้งเตือนผ่าน Microsoft Teams ที่ร้อยละ 100

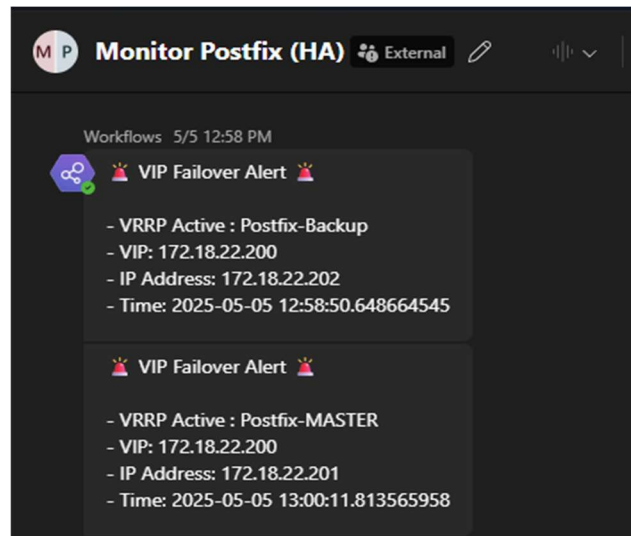


Figure 5. Alert Notification Test Results in Microsoft Teams

7. สรุปผลการวิจัย (Conclusion)

งานวิจัยนี้ประสบความสำเร็จในการพัฒนาสถาปัตยกรรมความพร้อมใช้งานสูงของระบบส่งต่ออีเมลปลอดภัยเพื่อปิดช่องว่างสำคัญของระบบแอปพลิเคชันดั้งเดิมที่ไม่รองรับการยืนยันตัวตนสมัยใหม่โดยผลการศึกษาตอบวัตถุประสงค์การวิจัยทั้ง 3 ข้อ ดังนี้

1. ด้านการแก้ปัญหาความเข้ากันไม่ได้ นวัตกรรม OAuth2 Agent ที่พัฒนาขึ้น สามารถทำหน้าที่เป็นตัวกลางในการแปลงโปรโตคอลได้อย่างสมบูรณ์ ส่งผลให้อีเมลจากระบบดั้งเดิมสามารถส่งผ่าน Microsoft 365 ด้วยมาตรฐาน OAuth 2.0 ได้สำเร็จร้อยละ 100% โดยไม่ต้องแก้ไขซอร์สโค้ดของแอปพลิเคชันต้นทาง

2. ด้านสมรรถนะระหว่างความปลอดภัยและประสิทธิภาพ ผลการทดสอบยืนยันว่าการเปลี่ยนผ่านสู่การยืนยันตัวตนสมัยใหม่ แม้จะเพิ่มความหน่วงเฉลี่ยเพิ่มขึ้นเป็น 2.158 วินาที เมื่อเทียบกับ 1.908 วินาที ในระบบส่งต่ออีเมลดั้งเดิม แต่เป็นต้นทุนที่คุ้มค่าเพื่อแลกกับการยกระดับความปลอดภัยจากการเลิกใช้รหัสผ่านถาวร ไปสู่การแอคเซสโทเคนที่มีการจำกัดเวลา ซึ่งช่วยลดความเสี่ยงจากการโจรกรรมข้อมูลประจำตัว (Credential Theft) ได้อย่างมีนัยสำคัญ

3. ด้านความต่อเนื่องของบริการ สถาปัตยกรรมความพร้อมใช้งานสูงที่นำเสนอสามารถกู้คืนระบบจากการล้มเหลวทั้งในระดับโหนดและระดับบริการได้ภายในเวลาเฉลี่ยต่ำกว่า 1.400 วินาที ซึ่งผ่านเกณฑ์น้อยกว่า 1.800 วินาทีที่ยืนยันถึงความพร้อมในการรองรับงานระดับองค์กรที่ต้องการความเสถียรสูง

8. อภิปรายผลการวิจัย (Discussion)

1. การเติมเต็มช่องว่างด้านการยืนยันตัวตน งานวิจัยในอดีตของ Sharabov et al. (2024) และ Oliveira et al. (2022) ได้พิสูจน์แล้วว่า Postfix เป็น MTA ที่มีความยืดหยุ่นสูงในการปรับแต่งเพื่อความปลอดภัย ในขณะที่ Gbenle et al. (2022) ได้เสนอแนวทางการใช้ Secure Gateway เป็นตัวกลางจัดการกระบวนการยืนยันตัวตน (OAuth 2.0) แทนระบบหลังบ้านเพื่อลดความซับซ้อนและความเสี่ยง งานวิจัยนี้จึงได้นำแนวคิดสถาปัตยกรรมตัวกลางดังกล่าวมาประยุกต์ใช้ร่วมกับกลยุทธ์ Encapsulation ตามแนวคิดของ Mishra (2020) ผ่านการพัฒนา OAuth2 Agent ให้ทำหน้าที่เป็นสะพาน (Bridge) เชื่อมต่อระหว่างโปรโตคอลเก่าและใหม่ ผลสำเร็จ 100% ในการส่งอีเมลยืนยันว่าองค์กรสามารถขยายอายุการใช้งานของแอปพลิเคชันดั้งเดิมได้ โดยไม่ต้องกระทบต่อการทำงานหรือแก้ไขซอร์สโค้ดเดิม

2. นัยสำคัญของสมรรถนะระหว่างประสิทธิภาพและความปลอดภัย ผลการทดสอบพบว่าค่าความหน่วงเฉลี่ยเพิ่มขึ้นเป็น 2.158 วินาที (เพิ่มขึ้น 0.25 วินาที) ซึ่งสอดคล้องกับข้อค้นพบของ Gbenle et al. (2022) ที่ระบุว่ากระบวนการจัดการโทเคนในสถาปัตยกรรมแบบกระจายศูนย์ย่อมมีต้นทุนทางเวลา (Overhead) อย่างไรก็ตาม งานวิจัยนี้ได้แก้ปัญหาดังกล่าวด้วยกลไกการแคชโทเคน ซึ่ง Gbenle et al. (2022) แนะนำว่าเป็นวิธีปฏิบัติที่ดีที่สุด ในการลดภาระการร้องขอไปยังเซิร์ฟเวอร์

และเมื่อพิจารณาพร้อมกับผลการศึกษาของ Matcha & Kumar (2025) ที่ระบุว่าการจัดเก็บโทเคนอย่างปลอดภัยด้วยการเข้ารหัสสามารถลดความเสี่ยงจากการโจรกรรมข้อมูลได้ถึงร้อยละ 88.6 งานวิจัยนี้จึงได้พัฒนากลไกการเข้ารหัส (Symmetric Encryption) เข้ากับระบบจัดเก็บโทเคนตามแนวทางดังกล่าว ซึ่งความหวังที่เพิ่มขึ้นเพียงเล็กน้อยนี้เป็นสิ่งที่คุ้มค่าและยอมรับได้ เพื่อแลกกับการกำจัดความเสี่ยงจากการใช้รหัสผ่านถาวรอย่างสิ้นเชิง

3. ประสิทธิภาพของการตรวจสอบระดับบริการ ในประเด็นความพร้อมใช้งานผลการทดลองชี้ให้เห็นข้อจำกัดของแนวทางดั้งเดิมในงานของ Putranto (2019) ที่ใช้ Keepalived ตรวจสอบเพียงระดับโหนด ซึ่งไม่เพียงพอการทำงานวิจัยนี้ได้บูรณาการแนวคิด Service Behavior Monitoring ตามกรอบของ Faseeha et al. (2025) มาพัฒนาเป็น Custom Health Check Script ส่งผลให้ระบบมีขีดความสามารถในการรักษาตัวเองได้อย่างแม่นยำ โดยสามารถตรวจจับภาวะของบริการล้มเหลวและสั่งกู้คืนระบบได้ภายในเวลาเฉลี่ย 1.300 วินาที ผลลัพธ์นี้เหนือกว่าระบบความพร้อมใช้สูงทั่วไปที่ปล่อยให้เกิดความล้มเหลวของบริการแฝง กล่าวได้ว่าเครื่องมือระดับบริการใช้ไม่ได้ ยืนยันว่าสถาปัตยกรรมที่นำเสนอมีความพร้อมสำหรับการใช้งานในระดับองค์กรที่ต้องการความต่อเนื่องทางธุรกิจสูง

9. ข้อเสนอแนะงานวิจัย (Recommendation)

จากผลการวิจัยและข้อสังเกตที่พบระหว่างการพัฒนา มีข้อเสนอแนะเพื่อให้เกิดประโยชน์ต่อการนำไปใช้งานจริงและการพัฒนาต่อยอดในอนาคต ดังนี้

1. การพัฒนาระบบแจ้งเตือนเชิงลึก ควรปรับปรุงระบบแจ้งเตือนบน Microsoft Teams ให้ระบุรหัสข้อผิดพลาด (Error Code) และแนบข้อมูล Log ล่าสุดไปกับข้อความ เพื่อช่วยให้ผู้ดูแลระบบวิเคราะห์สาเหตุที่แท้จริง (Root Cause Analysis) ได้รวดเร็วและแม่นยำยิ่งขึ้น
2. การขยายผลสู่ผู้ให้บริการอื่น (Universal IdP Support) ควรขยายขอบเขตการทดสอบไปยัง IdP รายอื่น เช่น Google Workspace หรือ Okta โดยปรับโครงสร้าง Config ของ OAuth2 Agent ให้ยืดหยุ่นต่อการกำหนดค่า Endpoint เพื่อยกระดับโซลูชันสู่ความเป็นมาตรฐานสากล
3. การศึกษาด้านความสามารถในการขยายตัว (Scalability) ควรศึกษาประสิทธิภาพการจัดการทรัพยากรของ Python Agent ภายใต้สภาวะที่มีการส่งข้อมูลหนาแน่น (High Concurrency) เพื่อรองรับการขยายตัวของจำนวนแอปพลิเคชันรุ่นเก่าในอนาคต
4. การตรวจสอบความมั่นคงปลอดภัย (Security Audit) ก่อนนำไปใช้งานจริง (Production) ควรดำเนินการประเมินความเสี่ยง (Vulnerability Assessment) อย่างเคร่งครัด โดยเฉพาะการทำ SAST เพื่อตรวจสอบคุณภาพโค้ด และ Dependency Check เพื่อป้องกันความเสี่ยงจากไลบรารีภายนอก

10. การเปิดเผยการใช้ปัญญาประดิษฐ์เชิงสร้างสรรค์ (Declaration of Generative AI)

งานวิจัยนี้ใช้ปัญญาประดิษฐ์เชิงสร้างสรรค์ โดยได้เลือกใช้เครื่องมือ Google Gemini เพื่อเป็นเครื่องมือช่วยตรวจสอบความถูกต้องทางภาษา ความครบถ้วนของรูปแบบการอ้างอิง และปรับปรุงโครงสร้างประโยคให้กระชับตามหลักวิชาการ นำมาช่วยตรวจสอบครอบคลุมทุกหัวข้อภายในบทความ เพื่อให้การนำเสนอเนื้อหา มีความสมบูรณ์ยิ่งขึ้น โดยไม่ได้ใช้ในการสร้างสรรค์เนื้อหาใหม่หรือวิเคราะห์ข้อมูลแต่อย่างใด ทั้งนี้ได้กำหนดคำสั่ง (Prompt) ในการใช้งานดังนี้ ช่วยตรวจสอบความถูกต้องทางภาษา การจัดลำดับหัวข้อ รูปแบบการอ้างอิง และปรับแก้ไขโครงสร้างประโยคให้มีความกระชับ อ่านเข้าใจง่าย ซึ่งภายหลังการประมวลผล จากนั้นผู้เขียนได้ทำการตรวจทานและคัดกรองผลลัพธ์ทั้งหมดด้วยตนเองอย่างละเอียด และขอยืนยันความรับผิดชอบต่อความถูกต้องของบทความนี้แต่เพียงผู้เดียว

11. การแสดงบทบาทผู้เขียน (CRediT Author Statement)

Pichit Thongdom: Conceptualization, Methodology, Software, Validation, Formal Analysis, Investigation, Resources, Data Curation, Writing-Original Draft, Visualization.

Tanun Jaruvitayakovit: Conceptualization, Writing-Review & Editing, Supervision, Project Administration.

All authors read and approved the final manuscript.

12. เอกสารอ้างอิง (References)

- Cassen, A., & Contributors. (2017). *Keepalived User Guide (Release 1.2.15)*. The Keepalived Project.
https://keepalived.readthedocs.io/_/downloads/en/stable/pdf/
- Chatterjee, N., Majumdar, S., Das, P. P., & Chakrabarti, A. (2024). *Tool Assisted Agile Approach for Legacy Application Migration*. Research Square. <https://doi.org/10.21203/rs.3.rs-3323311/v1>
- Cryptography. (2023). *Welcome to Pyca/Cryptography (Version 41.0.7)* [Computer Software].
Cryptography. <https://cryptography.io/en/41.0.7>
- Dovecot Team. (n.d.). *Postfix and Dovecot SASL*. Dovecot. https://doc.dovecot.org/2.3/configuration_manual/howto/postfix_and_dovecot_sasl/#howto-postfix-and-dovecot-sasl
- Faseeha, U., Jamil Syed, H., Samad, F., Zehra, S., & Ahmed, H. (2025). Observability in Microservices: An In-Depth Exploration of Frameworks, Challenges, and Deployment Paradigms. *IEEE Access*, 13, 72011-72039. <https://doi.org/10.1109/ACCESS.2025.3562125>
- Gbenle, T. P., Abayomi, A. A., Uzoka, A. C., Ogeawuchi, J. C., Adanigbo, O. S., & Odofin, O. T. (2022). Applying OAuth2 and JWT Protocols in Securing Distributed API Gateways: Best Practices and Case Review. *International Journal of Multidisciplinary Research and Growth Evaluation*, 3(5), 628-634. <https://doi.org/10.54660/IJMRGE.2022.3.5.628-634>
- Hardt, D. (2012). *The OAuth 2.0 Authorization Framework (No. RFC6749)*. Internet Engineering Task Force. <https://doi.org/10.17487/rfc6749>
- Hoffman, P. (2002). *SMTP Service Extension for Secure SMTP over Transport Layer Security (No. RFC3207)*. Internet Engineering Task Force. <https://doi.org/10.17487/rfc3207>
- Klensin, J. (2008). *Simple Mail Transfer Protocol (No. RFC5321)*. Internet Engineering Task Force. <https://doi.org/10.17487/rfc5321>
- Matcha, S., & Kumar, M. (2025). Enhancing Software Security with OAuth 2.0: Implementation Strategies and Vulnerability Mitigation. *Journal of Emerging Technologies and Innovative Research*, 12(3), e886-e902.
- Microsoft. (2025a). *Microsoft Identity Platform and OAuth 2.0 Client Credentials Flow*. Microsoft Learn. <https://learn.microsoft.com/en-us/entra/identity-platform/v2-oauth2-client-creds-grant-flow>
- Microsoft. (2025b). *Microsoft Authentication Library (MSAL) for Python*. Microsoft Learn. <https://learn.microsoft.com/en-us/entra/msal/python>
- Mishra, A. (2020). Legacy System Modernization: Effective Strategies and Best Practices. *International Journal of Leading Research Publication*, 1(3), 1-7. <https://doi.org/10.5281/zenodo.14769544>
- Oliveira, P. M., Vieira, M. B., Ferreira, I. C., Leite, J. P. R. R., Oliveira, E. M., Kuehne, B. T., Moreira, E. M., & Carpinteiro, O. A. S. (2022). *ABL: An Original Active Blacklist Based on a Modification of the SMTP*. arXiv. <https://doi.org/10.48550/arXiv.2208.10602>
- Postfix. (n.d.). *Postfix Architecture Overview*. <https://www.postfix.org/OVERVIEW.html>
- Putranto, E. A. (2019). *High Availability Server Implementation Using NGINX, Keepalived, and Varnish to Ease Web Server Access*. ResearchGate. <https://www.researchgate.net/publication/334451751>
- Python Software Foundation. (2025). *smtplib - SMTP Protocol Client*. <https://docs.python.org/3.12/library/smtplib.html#module-smtplib>
- Sharabov, M., Tsochev, G., Gancheva, V., & Tasheva, A. (2024). Filtering and Detection of Real-Time Spam Mail Based on a Bayesian Approach in University Networks. *Electronics*, 13(2), 374. <https://doi.org/10.3390/electronics13020374>

The aiosmtpd Developers. (2025). *The SMTP Module. Read the Docs*. Python. <https://aiosmtpd.aio-libs.org/en/latest/smtp.html>

The Exchange Team. (2024). *Exchange Online to Retire Basic Auth for Client Submission (SMTP AUTH)*. Microsoft Community Hub. <https://techcommunity.microsoft.com/blog/exchange/exchange-online-to-retire-basic-auth-for-client-submission-smtp-auth/4114750>