network

MDPI

Article

An IMAP Agent Framework for Extending Email Functionality in Outsourced Mail Services

Xiuyuan Chen [1], Tomoaki Tsutsumi [2], Rei Nakagawa [3], Yong Jin [4] and Nariyoshi Yamai [3,*]

[1] Department of Electronic and Information Engineering, Tokyo University of Agriculture and Technology, Koganei 184-8588, Japan; rockcxy@net.cs.tuat.ac.jp
[2] Institute of Humanities and Social Sciences, University of Tsukuba, Tsukuba 305-8577, Japan; tsutsumi.tomoaki.gn@u.tsukuba.ac.jp
[3] Division of Electrical Engineering and Computer Science, Institute of Engineering, Tokyo University of Agriculture and Technology, Koganei 184-8588, Japan; rnakagawa@go.tuat.ac.jp
[4] Center for Information Infrastructure, Institute of Science Tokyo, Meguro 152-8550, Japan; yongj@cii.isct.ac.jp
* Correspondence: nyamai@cc.tuat.ac.jp

Abstract

This paper presents an organization-managed IMAP Agent framework for extending email functionality in environments that rely on outsourced mail services. In this study, outsourced mail services refer to externally operated mailbox providers offering sufficiently scalable email infrastructures and standard IMAP interfaces, such as Gmail, Microsoft 365, and other commercial mailbox providers. In the proposed framework, IMAP Agents are operated within an organization, while user authentication continues to rely on existing institutional infrastructures such as Identity Providers (IdP) or Integrated Authentication Infrastructure (IAI). The IMAP Agent operates as a post-authentication processing component using credentials issued by these infrastructures, without modifying or intervening in the outsourced mail service itself. The framework enables organization-managed mailbox-side email processing without requiring administrative control over the mail server or dependence on provider-specific APIs. As a proof of concept, representative email-processing functions are implemented, including detection of suspicious messages based on header-level authentication information and automatic insertion of thread-consistent warning messages without altering the original email content. To evaluate the feasibility of the proposed framework, a prototype system was implemented using multiple containerized IMAP Agent instances. The experimental results showed that warning messages were typically appended within approximately 300 ms after message detection. Multi-container evaluations ranging from 1 to 100 concurrent IMAP Agent instances demonstrated low CPU overhead and approximately linear memory growth under idle-monitoring conditions, indicating the operational feasibility of deploying multiple IMAP Agent instances on a single host. These results suggest that the proposed framework can provide provider-independent and organization-managed extension of email functionality in outsourced mail environments through standard IMAP operations.

Keywords: IMAP Agent; outsourced mail services; email functionality extension; containerized deployment; organizational email management; Docker

Check for updates

Academic Editor: Andreas Kassler

Received: 30 April 2026
Revised: 5 June 2026
Accepted: 10 June 2026
Published: 12 June 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Email remains an essential communication infrastructure for academic institutions and enterprises, supporting daily collaboration, information exchange, and organizational

workflows. According to recent statistics, the number of global email users is expected to reach 4.48 billion in 2024 [1], underscoring the continued importance of robust and extensible email systems. Email relies on standardized protocols such as the Simple Mail Transfer Protocol (SMTP) [2], the Post Office Protocol Version 3 (POP3) [3], and the Internet Message Access Protocol (IMAP) [4]. Among them, IMAP [5] enables users to access and manage emails stored on external servers across multiple devices through standardized mailbox operations.

Today, many organizations rely on outsourced mail services provided by externally operated mailbox providers offering sufficiently scalable email infrastructures and standard IMAP interfaces, such as Google Workspace [6], Microsoft 365 [7], and other commercial email platforms. While some providers expose programmable APIs or administrative integration mechanisms, such interfaces are often provider-specific and may not support portable mailbox-side processing through standard IMAP operations. Consequently, organizations are often unable to implement organization-specific email-processing functions in a provider-independent manner without depending on vendor-specific APIs or requiring administrative control over the outsourced mail server itself.

User authentication in institutional environments is typically managed through an Identity Provider (IdP) or Integrated Authentication Infrastructure (IAI) [8]. These systems issue and verify user credentials and enable users to access outsourced services, including externally operated email platforms. However, IdP and IAI operate primarily at the authentication layer and do not directly provide organization-managed mailbox-side email processing functionality after authentication has completed. Therefore, even when organizations control authentication infrastructure, they remain limited in their ability to extend email-processing behavior within outsourced mail environments.

To address this limitation, this paper proposes an organization-managed IMAP Agent framework that enables extension of email functionality without requiring modification of outsourced mail servers. In the proposed framework, IMAP Agents operate as post-authentication processing components under organizational management using credentials issued through IdP/IAI-based authentication infrastructures. Importantly, the IMAP Agent does not intervene in the authentication process itself; rather, it performs mailbox-side processing exclusively through standard IMAP operations after authentication has completed.

IMAP Agents [9] maintain persistent connections with mail servers and support operations beyond those of traditional email clients, including real-time monitoring via IMAP IDLE, thread-consistent message insertion via IMAP APPEND, header-level message analysis, mailbox reorganization, and rule-based automatic processing. In the proposed framework, these capabilities can be independently applied to multiple user mailboxes by deploying separate IMAP Agent instances for each user account.

The primary novelty of this work lies not in a new email-processing algorithm, protocol, or phishing-detection methodology itself, but in a framework that enables organizations to flexibly introduce organization-managed mailbox-side processing functions in outsourced mail environments where the mail server cannot be modified and provider-specific APIs cannot be relied upon. The proposed framework is therefore positioned as an architectural and operational extension framework for outsourced mail environments rather than as a new detection or filtering methodology. To demonstrate the feasibility of the proposed framework, representative mailbox-side processing functions were implemented, including a DMARC/SPF/DKIM-based warning-message insertion mechanism and automatic thread-consistent warning insertion without altering the original email content. The DMARC/SPF/DKIM-based processing mechanism is implemented solely as a demonstration application of the proposed framework and not as a novel phishing-detection methodology.

In addition, multiple deployment experiments were conducted using containerized IMAP Agent instances representing different user mailboxes. The experimental results showed that warning messages were typically appended within approximately 300 ms after message detection. Multi-container evaluations ranging from 1 to 100 concurrent IMAP Agent instances further demonstrated low CPU overhead and approximately linear memory growth under idle-monitoring conditions.

These results suggest that the proposed framework enables provider-independent and organization-managed extension of email functionality in outsourced mail environments through standard IMAP operations without requiring administrative control over outsourced mail servers.

2. Related Technology

2.1. IMAP Agent

2.1.1. Internet Message Access Protocol

The Internet Message Access Protocol (IMAP) is a widely adopted standard protocol for retrieving and managing email messages stored on a mail server. Unlike the Post Office Protocol (POP3), which downloads and typically removes emails from the server, IMAP retains messages on the server and synchronizes mailbox states such as read/unread status, flags, and folder structures across multiple devices. This functionality makes IMAP particularly suitable for modern environments where users access email services from heterogeneous platforms and devices.

IMAP supports server-side mailbox management, selective retrieval of message contents, and preview of email headers, enabling efficient management of large numbers of messages while reducing unnecessary bandwidth consumption. These capabilities also support collaborative and distributed use cases, including shared organizational mailboxes and multi-device synchronization.

2.1.2. Architecture of the IMAP Agent

The IMAP Agent is a client-side processing system that extends the functionality of conventional email environments by performing email inspection, monitoring, organization, and rule-based processing through standard IMAP operations. Unlike conventional Mail User Agents (MUAs), which primarily provide user-side mailbox access functions through local email clients, the IMAP Agent continuously interacts with the mail server using IMAP and performs autonomous processing operations on mailbox contents. The IMAP Agent can perform operations such as message filtering, mailbox reorganization, thread-consistent message insertion, and rule-based automatic processing without requiring administrative privileges on the mail server. Standard IMAP operations can also be used to retrieve, rewrite, and replace existing messages through mailbox-level processing mechanisms [10]. This independence makes the IMAP Agent suitable for outsourced mail environments operated by externally managed mailbox providers supporting standard IMAP operations, where direct administrative control over the mail server is not available.

- Independent Mode
  In independent mode, the IMAP Agent operates as a standalone mailbox-processing component that communicates directly with the mail server on behalf of the user. The agent continuously monitors mailbox updates and autonomously performs processing tasks using IMAP operations. This mode is suitable for organization-managed deployment scenarios where mailbox-side processing must be performed independently from existing email clients. Because each IMAP Agent instance can be independently associated with a user account, multiple mailboxes can be processed simultaneously without requiring modification of the user's local email client configuration.

https://doi.org/10.3390/network6020039

- Proxy Mode
  In proxy mode, the IMAP Agent functions as an intermediary between the email client and the mail server. This configuration allows the agent to intercept and process email traffic transparently before forwarding communication to the email client. Proxy mode may support transparent mailbox-side processing without requiring modifications to user workflows; however, it may also introduce additional considerations related to communication latency, protocol compatibility, and client interoperability. Therefore, the present study focuses primarily on the independent deployment mode as a proof-of-concept organization-managed framework for outsourced mail environments.

Figure 1 illustrates the architecture of the IMAP Agent and the structural differences between independent mode and proxy mode. A key feature of the IMAP Agent is its ability to maintain persistent communication with the mail server and monitor mailbox updates in real time. By using the IMAP IDLE command, the IMAP Agent continuously listens for mailbox update notifications and can immediately detect newly arrived messages. If the mail server does not support the IDLE extension, similar behavior can be approximated through periodic NOOP commands [11]. These capabilities enable organization-managed mailbox-side processing in outsourced mail environments where server-side modification is not feasible. In particular, IMAP Agents can support extensible email workflows, organization-specific mailbox processing policies, and automated post-authentication operations using only standard IMAP functionality.

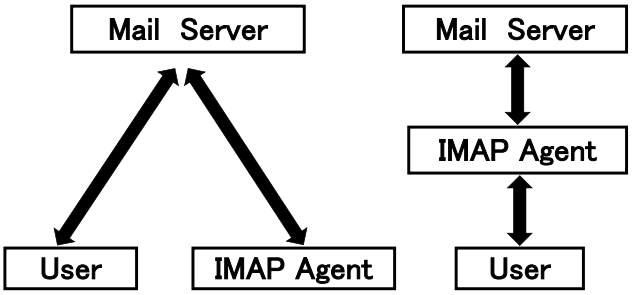


Figure 1. The Architecture of the IMAP Agent:Independent mode (**left**), Proxy mode (**right**).

2.2. Relation to Prior Work and Patents

A related patent application, JP 2018-5603 A [12], was filed in 2018 by a previous member of our research group. The patent describes an IMAP Agent capable of automated mailbox processing through IMAP-based operations such as message classification, relocation, and deletion. Although both the prior work and the present study utilize IMAP-based mailbox processing, the execution model and organizational assumptions differ significantly.

The primary distinction between the prior patent and the framework proposed in this study lies in the entity responsible for creating and operating IMAP Agent instances. In the prior patent, IMAP Agent instances were assumed to be created and managed by individual end users. In contrast, the framework proposed in this study assumes an organization-managed deployment model in which IMAP Agent instances are operated within an authentication infrastructure such as an IdP or Integrated Authentication Infrastructure (IAI) and function as post-authentication mailbox-processing components using credentials issued under organizational control.

Therefore, the primary novelty of this work lies not in IMAP-based mailbox processing itself, but in a framework that enables organizations to flexibly add their own mailbox-side email-processing functions in outsourced mail environments, where the mail server cannot

be modified and provider-specific APIs cannot be relied upon, by operating organization-managed IMAP Agents as post-authentication components.

Table 1 summarizes the differences between the prior IMAP Agent approach and the framework proposed in this study.

Table 1. Comparison between Previous IMAP Agent Approaches and the Proposed Framework.

Aspect	Previous IMAP Agent	Proposed Framework
Agent ownership	End user	Organization
Authentication context	User-managed	IdP/IAI-managed
Deployment target	Individual environment	Organizational environment
Processing method	IMAP-based	IMAP-based
Server modification required	No	No
Control structure	User-centric	Organization-managed

2.2.1. IMAP IDLE Command

The IMAP IDLE command [13] is a protocol extension that enables real-time mailbox update notifications between the mail server and the IMAP client. Once the client enters the IDLE state after mailbox selection, the server can immediately notify the client about mailbox updates such as arrival of new messages without requiring continuous polling from the client side. The IDLE command operates through the following sequence. After establishing a connection and authenticating to the IMAP server, the client issues a SELECT command to specify the target mailbox. The client then enters the IDLE state and waits for server notifications. When mailbox updates occur, the server sends update notifications such as EXISTS responses indicating the arrival of new messages. The client may then terminate the IDLE state, process the mailbox update, and subsequently resume IDLE operation if necessary. Using the IMAP IDLE command, the IMAP Agent maintains a persistent connection to the server, enabling seamless monitoring of incoming emails. This mechanism ensures that users are notified of important messages immediately, facilitating prompt responses and enhancing overall communication efficiency. The elimination of periodic polling also optimizes server performance and minimizes resource usage, making the IMAP IDLE command a cornerstone of modern email management. Figure 2 illustrates the protocol-level sequence of IMAP IDLE-based mailbox monitoring between the IMAP Agent and the outsourced mail service.

2.2.2. Related Works of IMAP Agent

Previous IMAP Agent studies [14] proposed mechanisms for maintaining continuous IMAP sessions and detecting mailbox updates through IMAP IDLE or periodic NOOP operations. Compared with traditional proxy-server architectures, IMAP Agent-based approaches provide a lightweight and mailbox-oriented mechanism for extensible email processing using standard IMAP operations. In addition, prior studies [15] compared IMAP with other email protocols such as POP3. Unlike POP3, which primarily downloads emails to a local client, IMAP retains mailbox contents on the server and supports synchronized mailbox management across multiple devices. IMAP also enables server-side folder management, mailbox synchronization, and remote message organization, making it suitable for modern distributed email environments where users require consistent mailbox access across heterogeneous platforms.

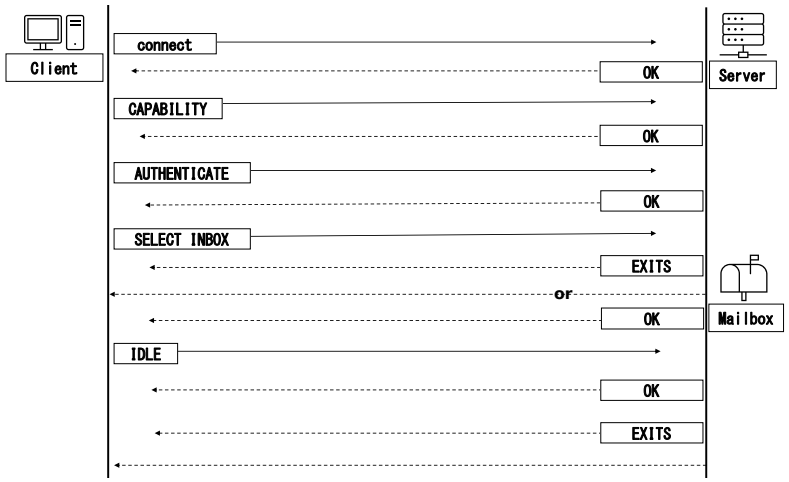


Figure 2. Example Protocol Sequence of IMAP IDLE-Based Mailbox Monitoring.

These characteristics make IMAP-based approaches particularly suitable for organization-managed mailbox-side processing in outsourced mail environments where server-side administrative control is unavailable.

2.3. Comparison with Existing Email Extension Approaches

Existing approaches for extending email functionality can generally be categorized into four types: server-side filtering frameworks such as Milter, server-side mailbox filtering approaches such as Sieve filtering, provider-specific APIs such as Gmail API or Microsoft Graph API, and client-side mailbox processing approaches such as local email client rules or plugins.

Milter (Mail Filter) [16] is a server-side framework designed to filter and process email messages during SMTP delivery. Because Milter requires integration with the Mail Transfer Agent (MTA), administrative privileges on the mail server are generally required. This dependency limits its applicability in outsourced mail environments where organizations cannot modify the server-side mail infrastructure.

Traditional server-side mailbox filtering approaches such as Sieve filtering [17] also provide rule-based email processing within the mail-server infrastructure. However, such approaches generally require provider-side support or integration into the mail-server environment. As a result, organizations relying on outsourced mail services cannot freely deploy or extend mailbox-side processing policies independently.

Some outsourced mail service providers expose programmable interfaces such as Gmail API or Microsoft Graph API, enabling mailbox access and integration with provider-managed services. However, these APIs are generally provider-specific and depend on vendor-defined interfaces, policies, and operational restrictions.

Conventional client-side approaches such as local email client rules or plugins can provide user-specific mailbox processing functions. However, these approaches are typically managed individually by end users and depend on specific email clients or user-side configurations. As a result, organizations cannot easily apply unified mailbox-side processing policies consistently across heterogeneous outsourced mail environments and multiple email clients.

In contrast, the framework proposed in this study relies exclusively on standard IMAP operations and performs mailbox-side processing after message delivery. As long as the outsourced mail service provides a standard IMAP interface, the proposed framework can operate in a provider-independent manner without requiring modification of the mail server itself. This portability is particularly important for organizational environments where multiple outsourced mail services may coexist or where vendor-independent

mailbox-side processing is desirable. Furthermore, the proposed framework enables organizations to flexibly introduce organization-managed mailbox-side processing functions even in outsourced mail environments where the mail server cannot be modified and provider-specific APIs cannot be relied upon.

Table 2 summarizes the differences between representative server-side, provider-dependent, and conventional client-side email extension approaches and the proposed IMAP Agent framework.

Table 2. Comparison between Existing Email Extension Approaches and the Proposed IMAP Agent Framework.

Aspect	Milter	Provider APIs	Proposed IMAP Agent
Deployment layer	Server-side	Provider API layer	Mailbox-side
Admin privilege	Required	API authorization	Not required
Dependency	Server dependent	Provider dependent	Provider independent
Protocol	SMTP	Proprietary APIs	IMAP
Applicable environment	Self-managed mail servers	Supported providers only	IMAP-compatible outsourced services
Portability	Moderate	Limited	High

2.4. Integrated Authentication Infrastructure

An Integrated Authentication Infrastructure (IAI) is designed to unify user identity management and authentication across multiple organizational services. Typical components include Single Sign-On (SSO) [8], Multi-Factor Authentication (MFA) [18], and OAuth-based authorization [19]. These mechanisms allow users to authenticate once and subsequently access multiple services without repeatedly entering credentials.

Figure 3 illustrates a conventional outsourced mail access flow without the proposed IMAP Agent framework. The user first attempts to access the outsourced mail service. Authentication processing is then delegated to the organization-managed IdP/IAI infrastructure, where operations such as SSO or MFA are performed. After successful authentication, mailbox access is granted by the outsourced mail provider.

Although authentication is centrally managed within the organization, mailbox-side behavior and processing remain controlled entirely by the outsourced mail provider. Consequently, organizations cannot directly extend or customize mailbox-side email-processing functionality after authentication has completed. In the proposed framework, IMAP Agents operate only after successful authentication has completed and use credentials issued within the organizational management domain. The IMAP Agent itself does not modify or intervene in the authentication infrastructure. A practical example of IAI deployment is the hybrid Identity-as-a-Service (IDaaS) system implemented at Tokyo University of Agriculture and Technology (TUAT), which integrates OAuth-based SSO and MFA across campus services [20]. This example illustrates that institutional authentication infrastructures can centrally manage user authentication for multiple services. However, such infrastructures do not directly provide provider-independent mailbox-side processing through standard IMAP operations. This limitation further motivates the proposed IMAP Agent framework as a post-authentication mailbox-processing mechanism operated under organizational control.

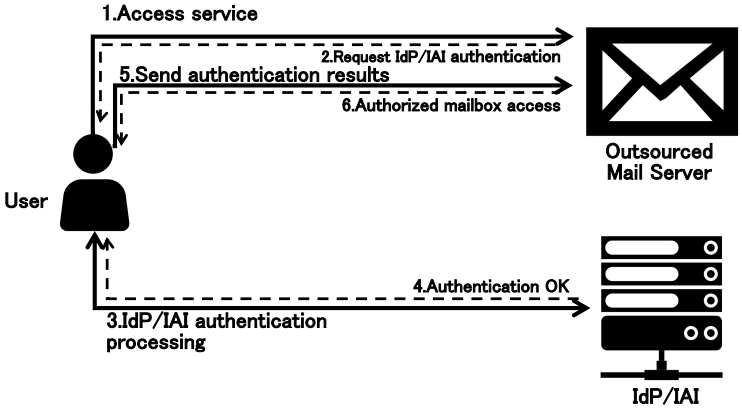


Figure 3. Conventional Organizational Email Access Flow in Outsourced Email Environments (Without IMAP Agent).

2.5. Container Technology

Container technology [21] provides lightweight execution environments that package applications together with their runtime dependencies. Unlike conventional virtual machines [22], containers share the host operating-system kernel while maintaining process-level operational separation. This design enables efficient deployment and execution of multiple independent application instances with relatively low resource overhead.

In this study, Docker is used as a practical implementation of container technology to deploy multiple independent IMAP Agent instances. Each container hosts one IMAP Agent associated with a separate user mailbox, enabling simulation of organization-managed multi-user deployment environments. The use of containers is not itself the primary contribution of the proposed framework. Rather, containers are used as a practical mechanism for deployment management, operational separation, and reproducible multi-user experimentation. Container isolation in this study is primarily intended to simplify deployment and operational management of independent IMAP Agent instances rather than to provide complete security isolation guarantees.

3. Proposed Framework

3.1. Motivation

Many organizations rely on outsourced mail services such as Google Workspace or Microsoft 365 while performing authentication through existing Identity Providers (IdP) or Integrated Authentication Infrastructure (IAI). Although these authentication systems allow users to access outsourced services securely, organizations generally do not have administrative control over the outsourced mail servers themselves. As a result, organization-managed mailbox-side processing functions—such as automated inspections, warning insertion, organization-wide mailbox policies, or extensible email workflows—cannot be implemented directly at the mail-server side.

Furthermore, while some outsourced mail providers expose programmable APIs, such interfaces are typically provider-specific and may not support portable mailbox-side processing across heterogeneous outsourced mail environments. Consequently, organizations may authenticate users through institution-managed infrastructures while still lacking direct control over mailbox-side email processing behavior after authentication has completed. To address this limitation, this paper proposes an organization-managed IMAP Agent framework that enables mailbox-side email functionality extension without requiring modification of outsourced mail servers. In the proposed framework, IMAP Agents operate as post-authentication mailbox-processing components using credentials issued through IdP/IAI-managed authentication infrastructures. The IMAP Agent itself

does not modify or intervene in the authentication infrastructure. Because IMAP Agents can independently process mailbox contents using standard IMAP operations, organizations can deploy organization-wide mailbox processing policies or selectively apply user-specific mailbox extensions after authentication has completed. Although IMAP Agents provide flexible mailbox-side processing capabilities, large-scale deployment and operational management of multiple agents may become operationally complex. Therefore, the proposed framework adopts containerization as a practical deployment mechanism. Each container encapsulates one IMAP Agent instance and its execution environment, enabling consistent deployment and operational management of per-user IMAP Agents across organizational environments. Credentials are provided to IMAP Agent instances at runtime under organizational management.

Figure 4 illustrates the limitations of conventional outsourced mail environments, where organizations may authenticate users through institutional authentication infrastructures but cannot directly extend mailbox-side functionality within the outsourced mail server.

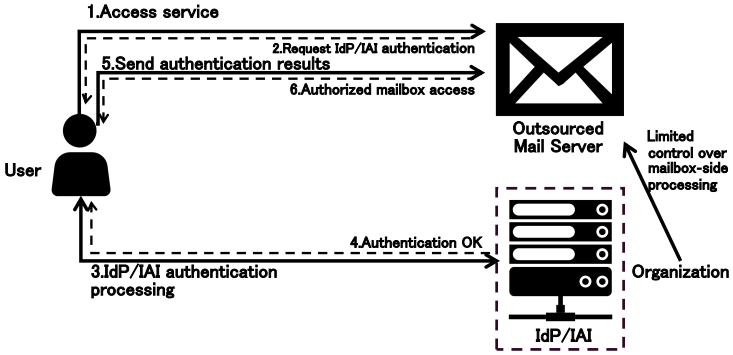


Figure 4. Limited extensibility in outsourced mail environments. Mailbox-side processing functions are dependent on provider-specific mechanisms and cannot be flexibly extended through standard organization-managed operations.

3.2. System Design

To enable organization-managed mailbox-side processing in outsourced mail environments, the proposed framework deploys IMAP Agents as lightweight post-authentication processing components operating independently from the outsourced mail server. Each IMAP Agent accesses mailbox contents through standard IMAP operations after successful user authentication has completed. In the proposed framework, IMAP Agents operate only after successful authentication through the organizational IdP/IAI infrastructure. Credentials used for mailbox access are assumed to be issued and managed within the organizational authentication domain and are provided to each IMAP Agent instance at runtime. The IMAP Agent itself does not intervene in, replace, or modify the authentication infrastructure. Each IMAP Agent instance operates independently and performs mailbox-side processing tasks such as mailbox monitoring, rule-based message inspection, thread-consistent warning insertion, and automated mailbox management. Because all processing is conducted through standard IMAP operations after message delivery, the proposed framework does not require modification of SMTP infrastructure, outsourced mail servers, or provider-side email systems.

The proposed framework assumes that the organizational authentication infrastructure and execution environment are managed within trusted organizational boundaries. Containerization is primarily used to simplify deployment management and operational separation of multiple independent IMAP Agent instances rather than to provide complete security isolation guarantees.

3.3. System Architecture

Figure 5 illustrates the overall architecture of the proposed organization-managed IMAP Agent framework. The authentication workflow between the user, outsourced mail server, and organizational IdP/IAI remains unchanged from the conventional outsourced mail environment shown in Figure 3. The user first accesses the outsourced mail service, which delegates authentication processing to the organizational IdP/IAI infrastructure. After successful authentication, mailbox access is authorized based on the authentication result issued by the IdP/IAI. In outsourced mail environments, organizations typically do not have administrative access to the mail server. Consequently, organization-managed mailbox-side processing functions cannot be implemented through direct server-side customization and must instead operate through external mailbox access mechanisms.

The proposed IMAP Agent framework addresses this limitation by operating as a post-authentication mailbox-processing component using standard IMAP operations. After authentication has completed, the IMAP Agent establishes an IMAP session with the outsourced mail server using credentials issued through the organizational IdP/IAI infrastructure. The IMAP Agent operates outside the authentication process itself and does not modify or replace existing authentication mechanisms. Each IMAP Agent instance operates independently within a user-side execution environment such as a local PC, virtual machine, or container. Through standard IMAP operations, the IMAP Agent performs mailbox-side processing tasks including message inspection, mailbox monitoring, and thread-consistent warning insertion. Because all processing occurs after message delivery through standard IMAP access, the proposed framework does not interfere with SMTP delivery procedures or outsourced authentication infrastructures.

This design enables provider-independent organization-managed mailbox-side processing in outsourced mail environments without requiring administrative privileges or server-side modification of the outsourced mail service.

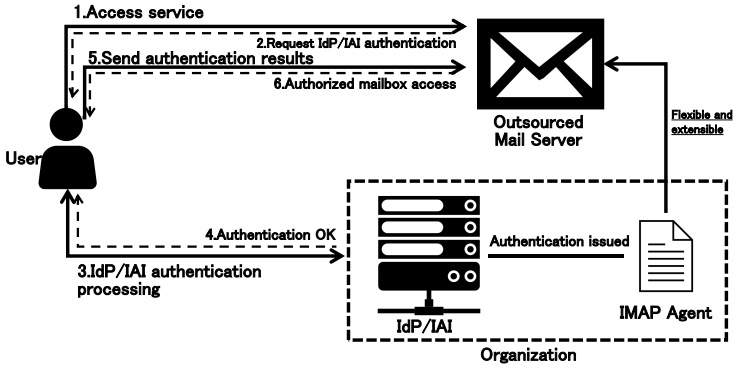


Figure 5. Proposed post-authentication IMAP Agent architecture. The Agent accesses the outsourced mailbox through standard IMAP operations after user authentication, enabling organization-managed mailbox processing without modifying the external mail server.

3.4. Credential Handling and Security Assumptions

In the proposed framework, IMAP Agent instances operate using credentials issued through existing organizational IdP/IAI infrastructures after authentication has completed. The present study assumes that authentication infrastructures and execution environments are operated within trusted organizational boundaries. In the current proof-of-concept implementation, credentials are dynamically provided to IMAP Agent containers at runtime in order to simulate post-authentication execution. The framework itself does not introduce a new authentication mechanism and instead relies on existing organizational authentication infrastructures such as SSO, MFA, and OAuth-based authentication systems. All mailbox

contents and credentials are transmitted over TLS-protected IMAPS connections (port 993). The present study does not assume permanent credential storage inside container images, and credentials are assumed to be held temporarily in process memory during execution. Credential rotation, revocation, and advanced token lifecycle management are outside the scope of the current implementation and remain future work.

Independent IMAP Agent containers are deployed separately for individual user accounts to simplify operational separation. However, the present study does not claim that containerization alone completely mitigates host-level compromise or privilege-escalation risks. The framework assumes that execution environments are operated within trusted organizational boundaries; therefore, host-level compromise or memory-dumping attacks are considered outside the trust boundary assumed by the present framework and are not claimed to be mitigated by container isolation alone.

3.5. Threat Model and Security Considerations

The proposed IMAP Agent framework was analyzed using the STRIDE threat model, a widely used threat-classification framework that categorizes security threats into Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege [23,24]. Spoofing risks are mitigated by relying on existing organizational IdP/IAI infrastructures for user authentication. Tampering risks are reduced because the framework appends warning messages without modifying the original email content. Repudiation risks are partially addressed because the IMAP Agent operates as an organization-managed background component rather than as an end-user-controlled client application. Mailbox-side processing operations performed through IMAP commands can therefore remain traceable through organization-side operational logs, while user authentication and login auditing are handled by the underlying IdP/IAI infrastructure.

Information disclosure risks are reduced because mailbox contents and credentials are transmitted through TLS-protected IMAPS connections. In addition, credentials are dynamically provided at runtime and are not assumed to be permanently stored inside container images. Denial-of-Service considerations include temporary IMAP communication failures and mailbox synchronization interruptions, for which reconnection mechanisms are supported. Elevation-of-Privilege risks are partially mitigated through operational separation using independent IMAP Agent instances. However, the present study does not claim comprehensive mitigation against host-level compromise or large-scale attacks. The purpose of this discussion is to clarify the operational assumptions and security boundaries of the proposed framework rather than to claim comprehensive mitigation of all security threats. Comprehensive security hardening and large-scale threat evaluation remain future work.

3.6. Practical Example of IMAP Agent Operation

The IMAP Agent operates as a per-user mailbox-side processing component that continuously monitors mailbox updates and performs organization-managed processing tasks after authentication has completed. Once a valid IMAP session has been established through credentials issued by the organizational IdP/IAI infrastructure, the IMAP Agent maintains continuous mailbox monitoring using the IMAP IDLE command. When a new message arrives, the IMAP Agent retrieves message metadata and evaluates header-level authentication information such as DMARC, SPF, and DKIM results. If the message satisfies organization-defined suspicious-message conditions, the IMAP Agent automatically generates an organizational warning message and appends it to the same mailbox thread using the IMAP APPEND command.

Importantly, the original email itself remains unchanged. Instead, the IMAP Agent inserts an additional warning message associated with the same conversation thread. To preserve thread consistency, the IMAP Agent maintains the original message thread relationship using standard email header fields while performing mailbox-side insertion through IMAP operations. All mailbox-side processing is performed independently within isolated IMAP Agent execution environments. This design enables organization-managed mailbox-side processing policies to be consistently applied across multiple user mailboxes without modifying outsourced mail servers or relying on provider-specific APIs. If temporary IMAP communication failures or network interruptions occur, the IMAP Agent can resume mailbox monitoring by re-establishing the IMAP session and continuing mailbox synchronization after reconnection.

Figure 6 illustrates the mailbox-side message-processing sequence for warning-message insertion performed by the IMAP Agent after suspicious-message detection.

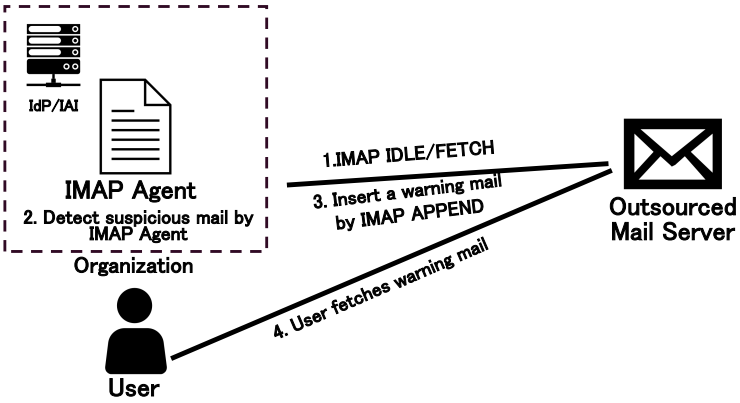


Figure 6. Message Processing Sequence for Warning-Message Insertion into the Original Email Thread.

4. Implementation and Evaluations for Proof of Concept

Building on the system design presented in Section 3.2, this chapter experimentally demonstrates the feasibility of deploying the proposed IMAP Agent framework within a simulated post-authentication environment as a proof of concept. The objective is to validate that the framework can provide organization-managed mailbox-side processing functions across multiple user accounts through containerized IMAP Agent deployment using standard IMAP operations. A representative use case involving DMARC-based suspicious-message detection and thread-consistent warning insertion is implemented to illustrate the extensibility of the proposed framework.

4.1. Deployment Objective and Architecture

Organizations that rely on outsourced mail services typically do not have administrative access to the mail server infrastructure. Consequently, organization-managed mailbox-side processing functions cannot be implemented through direct server-side customization.

To address this limitation, the proposed framework deploys independent IMAP Agent instances that access outsourced mailboxes using standard IMAP operations after authentication has completed. Figure 7 illustrates an example configuration in which multiple IMAP Agent instances are deployed for separate user mailboxes. Each IMAP Agent instance operates independently, establishes an IMAP session using credentials issued through the existing IdP/IAI infrastructure, and performs mailbox-side processing tasks such as mailbox monitoring, header inspection, and thread-consistent warning insertion. Because all processing is conducted through standard IMAP access after authentication, the framework does not require administrative privileges or modification of the outsourced mail server itself. To evaluate the feasibility of this approach, a multi-user deployment

environment was emulated using Docker containers, where each container hosts one independent IMAP Agent instance corresponding to a separate authenticated mailbox. In this setup, mailbox credentials were provided to each IMAP Agent instance at runtime to simulate post-authentication execution following authentication through an organizational IdP/IAI environment.

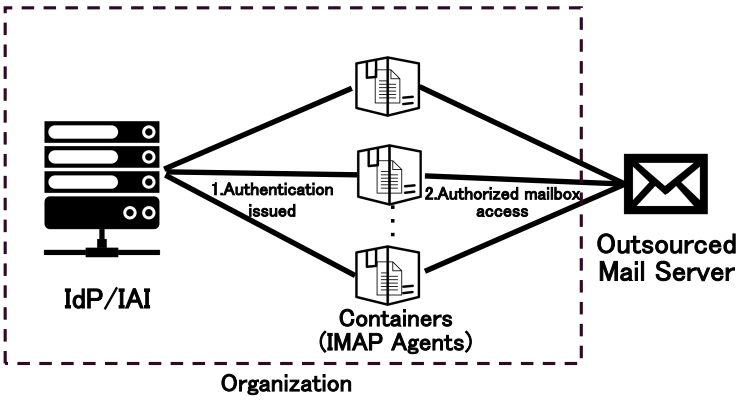


Figure 7. Deployment topology of multiple containerized IMAP Agent instances.

The deployment process consisted of the following steps:

1. Define user-specific mailbox configurations and runtime credentials.
2. Build a Docker image containing the IMAP Agent prototype and its dependencies.
3. Launch one container per mailbox using Docker runtime parameters.
4. Monitor runtime behavior through Docker Desktop and terminal logging.

To ensure reproducibility and clarity of the experimental setup, the implementation environment used in this study is summarized as follows:

1. The IMAP Agent prototype used in this experiment was implemented in C using libcurl for IMAP communication and IDLE-based message monitoring.
2. Host OS: macOS 26.4.1 [25].
3. Docker environment: Docker Desktop 4.70.0 [26].
4. IMAP server: GMO Onamae Mail Hosting (IMAPS, port 993) [27].

The same environment was used consistently across all the experiments, including multi-container deployment, warning-message insertion, and latency measurements. Containerization in this study was primarily adopted as a practical deployment mechanism to simplify operational management and reproducible execution of multiple independent IMAP Agent instances. The proposed framework assumes that authentication infrastructures and execution environments are operated within trusted organizational boundaries. Container isolation is used for operational separation rather than as a complete security isolation guarantee.

4.2. Demonstration of Extended Email Functionality: Header-Based Warning Injection

To demonstrate practical mailbox-side processing using the proposed framework, each IMAP Agent instance was equipped with a header-based authentication evaluation function. When a new message arrives, the IMAP Agent retrieves and analyzes authentication-related message headers such as `Authentication-Results`, `DKIM-Signature`, and `Received-SPF`. If these headers indicate a server-side DMARC/SPF/DKIM authentication failure, the IMAP Agent automatically generates an organization-managed warning message and appends it to the same mailbox thread using the IMAP APPEND command. Importantly, the original email content itself remains unchanged. Instead, the IMAP Agent inserts an additional warning message associated with the original message thread through standard

IMAP operations. Thread consistency is preserved by maintaining the relationship between the original message and the inserted warning message using standard email header fields. Figure 8 illustrates an example warning message inserted into the original email thread. Note that this work does not aim to propose a novel phishing-detection methodology. The DMARC/SPF/DKIM-based processing mechanism is implemented solely as a demonstration application of the proposed framework, showing that organization-managed mailbox-side processing can be realized through standard IMAP operations.

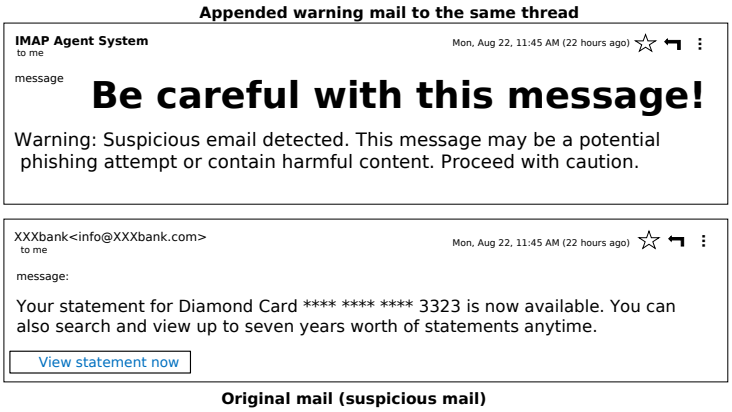


Figure 8. Example Warning Message Injected into the Original Email Thread.

To evaluate the responsiveness of the proposed IMAP Agent framework during active mailbox processing, an additional latency measurement experiment was conducted. The experiment measured the elapsed time between detection of a newly arrived message through IMAP IDLE notification and successful insertion of the warning message into the same mailbox thread using the IMAP APPEND command. Thirty independent measurements were conducted under the same experimental environment described previously. Table 3 summarizes the observed insertion latency, which ranged from 262 ms to 289 ms, with an average latency of approximately 274 ms. The experimental results indicate that warning messages can be appended to the mailbox thread with sufficiently low operational delay after suspicious-message detection. In practical operation, the warning message is typically delivered to the user nearly simultaneously with the original email notification, and no noticeable operational delay was observed during the experiments. These results suggest that the proposed framework can provide warning insertion with sufficiently low processing delay for practical mailbox-side operation while maintaining low operational overhead.

Table 3. Latency evaluation of warning-message insertion.

Metric	Measured Value
Number of measurements	30
Average latency	274 ms
Minimum latency	262 ms
Maximum latency	289 ms

Overall, the experiments demonstrate that the proposed IMAP Agent framework can provide organization-managed mailbox-side processing functionality in outsourced mail environments with low operational overhead and sufficiently low processing delay for practical warning-message insertion through standard IMAP operations.

4.3. Performance Evaluation Under Multi-Container Deployment

To evaluate the operational feasibility of the proposed framework, experiments were conducted using multiple independent IMAP Agent containers, each associated with a

separate mailbox account. During the experiments, all IMAP Agent instances continuously monitored mailbox updates using the IMAP IDLE command under idle-state conditions.

Table 4 summarizes the observed CPU and memory consumption for different numbers of concurrently operating IMAP Agent containers.

The experimental results showed that memory usage increased approximately linearly as the number of IMAP Agent instances increased. Each IMAP Agent instance consumed approximately 3 MB of memory, indicating low memory overhead even under multi-container deployment conditions. CPU utilization also remained extremely low throughout the experiments. Even in the 100-container deployment environment, total CPU utilization remained below 1 percent.

Although IMAP IDLE is designed as a waiting-state protocol extension, small amounts of CPU utilization were still observed due to connection maintenance and periodic internal processing within the IMAP client environment.

These results suggest that multiple independent IMAP Agent instances can coexist on a single host with low operational overhead under idle-monitoring conditions. Because outsourced mail services such as Gmail or Microsoft 365 already provide scalable IMAP infrastructures, the purpose of the present experiment is not to benchmark the scalability of the outsourced mail provider itself, but rather to demonstrate the operational feasibility of organization-managed multi-user IMAP Agent deployment as a proof of concept.

Table 4. Resource consumption under different numbers of IMAP Agent containers during idle-state monitoring.

Number of Containers	CPU Usage (%)	Memory Usage (MB)	Average Memory per Agent (MB)
1	0.01	3.14	3.14
5	0.05	15.68	3.14
10	0.10	31.34	3.13
20	0.12	62.77	3.14
50	0.52	160.06	3.20
100	0.95	313.52	3.14

5. Conclusions and Future Works

This paper presented an organization-managed IMAP Agent framework for extending mailbox-side email functionality in outsourced mail environments without requiring modification of outsourced mail servers. Unlike conventional server-side approaches such as Milter or provider-specific API integrations, the proposed framework performs post-authentication mailbox-side processing exclusively through standard IMAP operations, enabling provider-independent deployment across heterogeneous outsourced mail services.

The primary contribution of this work is not the introduction of a new email-processing algorithm, protocol, or phishing-detection methodology itself, but a framework that enables organizations to flexibly add their own mailbox-side email-processing functions in outsourced mail environments, where the mail server cannot be modified and provider-specific APIs cannot be relied upon, by operating organization-managed IMAP Agents as post-authentication components.

As a demonstration application of the proposed framework, a DMARC/SPF/DKIM-based warning-message insertion mechanism and thread-consistent warning-message insertion function were implemented using standard IMAP operations. The DMARC/SPF/DKIM-based processing mechanism is intended solely as a demonstration of organization-managed mailbox-side processing and not as a novel phishing-detection methodology. The experimental results demonstrated that warning messages could typically be appended

within approximately 300 ms after message detection. Additional multi-container deployment experiments ranging from 1 to 100 concurrent IMAP Agent instances showed low CPU utilization and approximately linear memory growth under idle-monitoring conditions, indicating the operational feasibility of deploying multiple IMAP Agent instances on a single host.

At the same time, the present study has several limitations. The current evaluation primarily focuses on proof-of-concept validation under idle-monitoring conditions and does not fully evaluate large-scale active workloads, long-term operational stability, or provider-specific behavioral differences across heterogeneous outsourced mail environments. In addition, the proposed framework assumes that authentication infrastructures and execution environments are operated within trusted organizational boundaries.

Future work will focus on extending the evaluation toward larger-scale organizational environments, including long-term operational testing, failure recovery behavior, and active mailbox-processing workloads. We also plan to investigate compatibility with modern authentication mechanisms such as OAuth2-based delegated authorization and evaluate interoperability across multiple outsourced mail providers including Gmail and Microsoft 365. In addition, future work will investigate proxy-mode IMAP Agent architectures and evaluate their operational characteristics, including communication latency, protocol compatibility, and client interoperability in transparent mailbox-processing scenarios.

Author Contributions: Conceptualization, T.T., Y.J. and N.Y.; methodology, X.C., T.T., R.N., Y.J. and N.Y.; software, X.C.; validation, X.C.; investigation, X.C.; data curation, X.C.; writing—original draft, X.C.; writing—review and editing, T.T., R.N., Y.J. and N.Y.; visualization, X.C.; supervision, T.T., R.N., Y.J. and N.Y.; project administration, N.Y.; funding acquisition, N.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Oberlo. Email Usage Statistics 2024. Available online: https://www.oberlo.com (accessed on 4 June 2026).
2. Klensin, J. Simple Mail Transfer Protocol. RFC 5321. 2008. Available online: https://www.rfc-editor.org/info/rfc5321 (accessed on 4 June 2026).
3. Gellens, R.; Newman, C.; Lundblade, L. POP3 Extension Mechanism. RFC 2449. 1998. Available online: https://www.rfc-editor.org/info/rfc2449/ (accessed on 4 June 2026).
4. Melnikov, A.; Leiba, B. Internet Message Access Protocol (IMAP)—Version 4rev2. RFC 9051. 2021. Available online: https://www.rfc-editor.org/info/rfc9051/ (accessed on 4 June 2026).
5. Crispin, M. Internet Message Access Protocol—Version 4rev1. RFC 3501. 2003. Available online: https://www.rfc-editor.org/info/rfc3501/ (accessed on 4 June 2026).
6. Google. Gmail API. Available online: https://developers.google.com/workspace/gmail (accessed on 4 June 2026).
7. Microsoft. Microsoft Graph API for Outlook Mail. Available online: https://learn.microsoft.com (accessed on 4 June 2026).
8. Bazaz, T.; Khalique, A. A review on single sign-on enabling technologies and protocols. *Int. J. Comput. Appl.* **2016**, *151*, 18–25. [CrossRef]
9. Yokogi, K.; Yamai, N.; Wang, J.; Kitagawa, N. Additional Services of Email Enabled by IMAP Agent. In *Proceedings of the DICOMO2018 (Symposium on Multimedia, Distributed, Cooperative and Mobile Systems), Awara, Japan, 4–6 July 2018*; Processing Society of Japan: Tokyo, Japan, 2018; pp. 1403–1406. Available online: https://ipsj.ixsq.nii.ac.jp/records/193635 (accessed on 4 June 2026). (In Japanese)
10. Myers, J. IMAP4 UIDPLUS Extension. RFC 2359. 1998. Available online: https://www.rfc-editor.org/info/rfc2359/ (accessed on 4 June 2026).

11. Daboo, C. Use of SRV Records for Locating Email Submission/Access Services. RFC 6186. 2015. Available online: https://www.rfc-editor.org/info/rfc6186/ (accessed on 4 June 2026).
12. Yamai, N.; Kitagawa, N.; Yokogi, K. E-Mail Processor and Programs. Japan Patent JP6757028B2, 16 September 2020. Available online: https://patents.google.com/patent/JP6757028B2/ja (accessed on 4 June 2026).
13. Leiba, B. IMAP4 IDLE Command. RFC 2177. 1997. Available online: https://www.rfc-editor.org/rfc/rfc2177 (accessed on 4 June 2026).
14. Crispin, M. Internet Message Access Protocol—Version 4rev1. RFC 2060. 1996. Available online: https://www.rfc-editor.org/info/rfc2060/ (accessed on 4 June 2026).
15. Tashiro, K.; Yamai, N.; Naoya, K. Flexible Email Forwarding System Using IMAP Agent Without Changing Server Configuration. In *Proceedings of the DICOMO2019, Koriyama, Japan, 3–5 July 2019*; Processing Society of Japan: Tokyo, Japan, 2019; pp. 1795–1800. Available online: https://ipsj.ixsq.nii.ac.jp/records/202534 (accessed on 4 June 2026). (In Japanese)
16. Mulligan, G. *Removing the Spam: E-Mail Processing and Filtering*; Addison-Wesley: Boston, MA, USA, 1999.
17. Guenther, P.; Showalter, T. Sieve: An Email Filtering Language. RFC 5228. 2008. Available online: https://www.rfc-editor.org/info/rfc5228/ (accessed on 4 June 2026).
18. IBM. What Is Multi-Factor Authentication? Available online: https://www.ibm.com/topics/multi-factor-authentication (accessed on 4 June 2026).
19. OAuth.net. OAuth 2.0. Available online: https://oauth.net/2/ (accessed on 4 June 2026).
20. Nemoto, T.; Mishima, K.; Aoyama, S. Implementation and initial operation of IDaaS as integrated authentication infrastructure in TUAT. In Proceedings of the SIGUCCS '23: Proceedings of the 2023 ACM SIGUCCS Annual Conference, Chicago, IL, USA, 26–29 March 2023; pp. 48–52.
21. Docker Inc. Docker Engine. Available online: https://docs.docker.com/engine/ (accessed on 4 June 2026).
22. Smith, J.E.; Nair, R. *Virtual Machines*; Morgan Kaufmann: Burlington, MA, USA, 2005. Available online: https://www.sciencedirect.com/book/9781558609105/virtual-machines (accessed on 4 June 2026).
23. Kohnfelder, L.; Garg, P. The Threats to Our Products. Microsoft Interface. 1999. Available online: https://shostack.org/files/microsoft/The-Threats-To-Our-Products.docx (accessed on 4 June 2026).
24. Shostack, A. *Threat Modeling: Designing for Security*; Wiley: Indianapolis, IN, USA, 2014.
25. Apple Inc. Mac. Available online: https://www.apple.com/ (accessed on 4 June 2026).
26. Docker Inc. Docker Desktop. Available online: https://www.docker.com/ (accessed on 4 June 2026).
27. GMO Internet Group, Inc. Onamae Server. Available online: https://www.onamae.com/ (accessed on 4 June 2026).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.