# A Flexible Role Based Secure Messaging System: An Information System Perspective

**Ositanwosu Obiajulu E.[1], Ajonuma Michael E.[2]**
[1]Department of Computer Science,
Nnamdi Azikiwe University Awka, Anambra State,
Nigeria. +234-806-025-3029
[2]Department of Computer Science,
Federal Cooperative College Oji, Enugu State,
Nigeria. +234-806-471-1605
[1]oe.ositanwosu@unizik.edu.ng,
[2]ajonumamichael@gmail.com
Corresponding emails: ajonumamichael@gmail.com
phone: +234-806-471-1605
https://doi.org/10.56201/ijcsmt.vol.12.no10.2026.pg63.73

***Abstract***
*In this research, we presented the construction of a flexible role-based secure managing service system which can operate in an open environment. The tool is aimed at providing protection of messages across domains, and caters to most of the commercial requirements for information integrity. The challenge of an unsecure messaging system was addressed by the construction of a flexible secure messaging service system, where role-based messaging and access control and messaging/permissions are managed based on roles, while authorization and encryption (cipher) keys are sent to registered phone numbers. The Software Development Life Cycle was adopted to carry out a detailed study of the existing mailing service systems and role-based messaging systems with a view to identifying the problems and loopholes in the secure role-based mailing system. Analysis and design of the new system were performed using popular software design tools. This study also contributes to providing security features based on architecture design, and demonstrates the implementation of information confidentiality and privacy.*

***Keyword****: Messaging system, Information system, Software Development Life Cycle*

## Introduction

Modern societies and organizations are more complex, dynamic, and flexible. People's roles, rights and duties change because of frequent re-organizations and evolutions of the market and customers' needs. Information is continuously generated and exchanged between all the involved parties, including confidential and private information. This information has to be protected against unauthorized access. Messaging systems like email systems convert information into predefined formats and deliver the information from senders to receipts [1]. Replay of the information in the course of delivery may be necessary. Messaging systems have been developed into one of the most commonly used electronic communication systems and have been applied to wider areas than those for which they were designed. Also, messaging systems are widely used to enable communications between people in diverse applications, such as businesses and health

systems [22]. In the setting of organizations, a message is sent to a person with the assumption that the receiver is the person who is responsible for dealing with the message's contents. To some degree, it requires that senders know about the identities of people corresponding to specific duties and issues, and that they can always get-to-date information on the relationships between issues, duties and the people who are responsible for them.

From a business point of view, the destination of these messages is organizational roles instead of the physical occupants of roles. A role is a position or function of an organization, associated with zero or multiple people who are usually authorized to perform certain transactions. People are related to the duties and tasks by being assigned as occupants of the corresponding role. Role assignment is dynamic, and should be instantaneous, especially when a role is being removed from a current role occupant. A role is an abstract model in an organization [24], which specifies a set of duties and can be associated with a set of people in the organization. Security is required when the information being carried by a message is important and/or confidential. Message security means various things, including: confidential message contents have integrity and are protected from being modified during the course of their transportation; recipients cannot falsely deny having received messages that have been delivered; and the identity of message senders can be verified at any time after the message has been sent.

Secure messaging solutions based on traditional cryptographic systems and public key infrastructure show their limitations in dynamic contexts, in terms of flexibility and manageability. One significant impediment to the widespread of public-key cryptography is its dependence on a public-key infrastructure that is shared among its users. Before secure communications can take place, both sender and receiver must generate encryption and signature key pairs, submit certificate requests along with proof of identity to a Certificate Authority (CA), and receive CA-signed certificates, which they can then use to authenticate one another and exchange encrypted messages. This process can be both time-consuming and error-prone, and is especially prohibitive for novice computer users. Frequently, individuals who can already receive encrypted email are still unable to send secure messages to others due to lack of preparedness, limited interoperability, device limitations, or lack of technical competence on the receiving end. The role-based access control model (RBAC) [3] has attracted a huge amount of research interest in the last decade [27,4,5,26]. The general idea of the RBAC model is that permissions are grouped by roles and users are assigned role memberships to acquire permissions allocated to the roles. Roles can be linked through various relations, including role hierarchy relations, constraints such as role conflict, and so on. These relations allow permissions grouped by roles to be derived or excluded according to the definitions of the relations.

This thesis is to address the software engineering issues related to the development of the architecture for the secure role-based messaging model proposed in [6]. This research advances the state of the art in secure role-based messaging in the following ways. Firstly, it presents a construction model that enables the extension of current messaging systems without invalidating their compliance with existing standards. Secondly, it implements the proposed secure role-based messaging model incrementally, so that either senders or receivers or both may implement the model independently. More specifically, the thesis demonstrates how to extend current messaging systems to achieve secure role-based messaging while keeping compliance with existing standards and enabling easy integration with existing systems. The following specific objectives are fulfilled in this research; to achieve the integration of existing systems when providing a secure messaging system, to design an automatic password generator and user access code mechanisms that will be sent to users' mobile phones directly, and to design security into

the system architecture that ensures the privacy and confidentiality of the content of mail exchanged within the organization.

**Related Works**

At this point, we review the current progress on the subject of message security. The topics covered by this review include messaging standards, messaging security, policy based management, commercial security policies, security infrastructures, the RBAC model, and other cryptographic related research.

On internet messaging, X.400 Recommendations [7], which is a messaging standard, and internet email are the two most important messaging standards. The X.400 Recommendations were first published in 1984, followed by two substantial revisions in 1988 and 1992 respectively. Internet email stems from ARPANET's experimental text message communication system, and is also considered to be a simplified version of X.400. The Internet email was initially defined as Advanced Research Project Agency (ARPA) Internet Text Messages Format (RFC822) and has been redefined by Internet Messages Format (RFC 2822) to facilitate the conveyance of content other than ANSI characters. A multipurpose internet mail extension (MIME, RFC 2045) [12] has been proposed to wrap different types of content and to incorporate this content into messages defined by RFC 2822. Klensin [17] proposed four main protocols: the Simple Message Transfer Protocol, the message Submission Protocol, the internet message Access Protocol and the Post Officer Protocol [25]. There are two main formats: the Internet Message Format and the Multipurpose Internet Message Extension [17].

In architecture, the internet email systems (IESs) bear some semblance to the X. 400 messaging system. The former has the following elements: Message User Agents (MUA), Message Transfer Agents (MTAs), Message Submission Agents (MSAs), Message Transfer Agents (MTAs), Message Submission Agents (MSAs), and Message Stores (MSs). On the protocols of the IESs, Myers (1996) wrote two essential protocols, which include the Post Protocol version 3 (POP3) and the Simple Mail Transfer Protocol (SMTP) [17]. The message Submission Protocol SMTP service extensions [18] and the Internet Message Access Protocol (IMAP) [8] are optional protocols that provide extra functionality. The Post Office version 3 (POP3) [25] defines the interoperations between a MUA and a MS, which allows the MUA to manipulate messages stored on the MS. SMTP was first defined by RFC 821 [28], and has been further developed by RFC2821 [17]. The message submission protocol, defined by RFC 2476 [14], is a protocol for messaging submission from UAs to MSAs. MUAs ensure messages contain all the necessary information and are constructed in the right format before they are submitted to MTAs. The Internet Message Access Protocol Version 4revl (IMAP4) [8] allows clients (the MUA) to access and manage messages stored on a server (the MS) as well as to manipulate mailboxes on the server. Messages in the internet email system are mainly in the format defined jointly by ARPA internet text message (RFC 822) [9]. The Internet Message Format (RFC2822) and Multipurpose Internet Mail Extensions (RFC 2045) [12].

The Multipurpose Internet Mail Extension (MIME) was later defined to extend the Internet Mail Format defined by RFC 822 and RFC2822 to accommodate message bodies that contain character sets other than ASCII, non-textual message content, multi-part message bodies, and textual headers not in ASCII. X.400 provides users with security services including labeling, secure access management, origin authentication, data integrity, data confidentiality, non-repudiation, and security management. Secure/Multipurpose Internet Mail Extensions (S/MIME) integrates Cryptographic Message Syntax (CMS) with MIME [11]. CMS is used to wrap MIME

objectives into CMS objects to provide enveloped data types and signed data types. With S/MIME, a message can be signed, or encrypted, or any combination of signing and encrypting. S/MIME was devised based on the X.509 public key infrastructure. A signed receipt may be requested for a signed-data object in a S/MIME message as soon as the recipient returns the signed receipt, the sender can verify the integrity of the message delivered to the recipient and the authenticity of the recipient.

There are other secure messaging protocols. Mont et al [24] provide secure messaging from the perspective of roles in a healthcare environment. Roles are employed as the basic unit of authorization. Users are authorized to perform operations on messages on behalf of roles if they hold the corresponding roles and the roles are allowed to perform the requested operations. Hasselbach [15] designed a system to provide secure mailing lists, providing secure messaging for group communication. The architecture design prevents information disclosure by separating the message processing into several independent components. Woltusen [30] imposed mandatory security processing at the operating system level, where all messages are modified when they pass through the operating system. The Trusted Email Open Standard (TEOS) [19] established a three-level identity environment for SMPT e-mail based on signatures, assertions, and certifications. Allman et al. [2] proposed an e-mail authentication system designed to verify the domain name system (DNS) domain of an e-mail sender. Here, the message integrity, by verifying the DNS domain of the sender and the message integrity, an email can be affirmed to be trusted in its sender's identity and the content's integrity. Sender ID [20] devised an authentication system to verify email origins. Wong and Schlitt [31] proposed the Sender Policy Framework (SPF), an extension to the SMTP. SPF allows the owner of an internet domain to use special formats of DNS TXT records to specify which machines are authorized to transmit e-mail for that domain. Lyon [21] defines a new identity associated with an e-mail message, called the Purported Responsible Address (PRA), which is determined by inspecting the header of the message. Several standards, including X.400 [7] and S/MIME [29,16] and OpenPGP [10-11] etc., have been designed to provide secure messaging. But the focus of these standards has primarily been on the issues of authentication, confidentiality, non-repudiation, integrity, and so on. The above-mentioned standards assume a closed environment for communication, thus they lack the capability of handling trust and access control in a distributed and open environment, where new member systems and clients join and leave dynamically without any pre-distributed knowledge of each other. Therefore, mechanisms for access control and trust handling in distributed and open environments are needed. There is very little effort on the authentication and access control aspects of secure messaging, and this is the main focus of the research reported herein.

**Methodology**

The methodology used for the study is the Software Development Life Cycle [23]. The method involves the following phases: requirement gathering, analysis, design, coding, testing, installation and deployment. Using this methodology, we developed a framework for a flexible role-based secure messaging system, which is flexible enough to cater for different security and messaging requirements whilst having the least impact on existing mailing systems and standards. Requirement Collection and Analysis to fulfill this stage, we analyzed the existing operational procedures in order to elicit the weaknesses therein. In general, a messaging system contains user agents to enable users to compose, send and receive messages, and a message submission agent to validate the user's submissions. A message delivery agent would transmit

messages while a message store will be responsible for storing and providing message access for users. Figure 1 shows a generic architecture for messaging systems. The Message User Agent (MUA) is the component that provides users with the facilities to compose, send and receive messages. It mediates communication between users and the other components. The Message Submission Agent (MSA) receives composed messages from MUAs, validates them, inserts compulsory or missing fields, makes necessary modifications to messages and then forwards them to the Message Transfer Agent (MTA). The MTA is responsible for transferring messages. It receives messages from a local MUA and/or MSA or a remote MTA in other systems. It delivers messages to other systems if the destination of the messages is not a local address. Two message store protocols are standardized for use on the internet-the Post Office Protocol Version 3 (POP3) and the Internet Message Access Protocol (edition 4) (IMAP4). IMAP4 is a more sophisticated protocol than POP3, and only IMAP4 will be considered in this research (although the same principles can be applied to POP3). The MS stores manage messages over the IMAP4 and POP3 protocols.

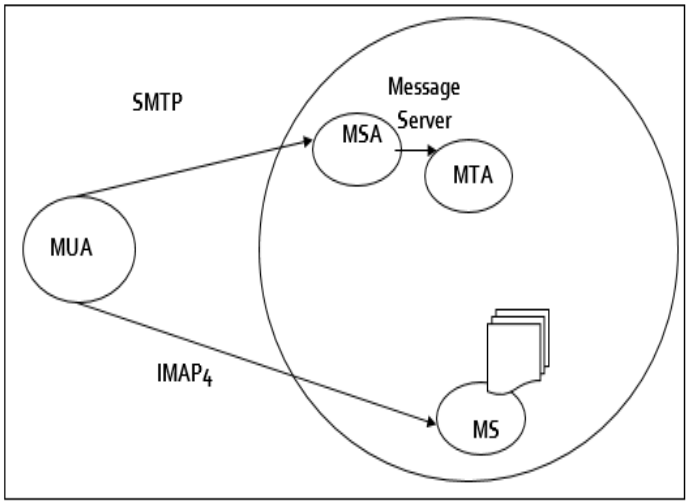



Figure 1: The Generic Architecture for Messaging Systems [6]

Of the challenges encountered within a large and structured organization, people often want to send a message to a given role within that organization, rather than to a specific individual. Unfortunately, most email systems are person to person and do not inherently support roles. The gaps between existing messaging systems and applications are exaggerated in large organizations, which have complex organizational structures and people continually adopt different roles during their professional life.

Two of the current ways of supporting role-based messaging are using dedicated role mailboxes and using distribution lists. When using dedicated role mailboxes, role occupants share a role mailbox by sharing the role password or role private key to access the role occupants are frequent. Furthermore, it is difficult to stop someone from accessing a role mailbox after the role has been removed from him or her without changing the role password. Sharing role-private keys is a better option if the keys are held in hardware tokens. The tokens can then be physically removed. But if the private key is held in an encrypted file, it is little better than sharing a role password, since the file can be copied at will. When distribution lists are used, messages sent to a role are copied to each role occupant. One of the drawbacks of this mode is that it does not cater to the temporal nature of role occupancy without significant management overhead. Removed

role occupants will continue to have access to the messages that were delivered to the role prior to their removal, while newly added role occupants will not be able to access those messages that were delivered to the role before their role assignment was made.

The issue is further complicated when secure messaging is required. Messages sent by a role need to be signed by a role, whilst encrypted messages received by a role need to be read by all role occupants. Role occupants could be given copies of the role's private key(s), but the management overhead of these keys, both by the individual and by the administrator, would be prohibitive. Enhancing the distribution list expansion method to re-encrypt the messages for each individual role occupant does not allow new role occupants to access the messages that have already been delivered, nor does it cater to role occupants whose role assignments are revoked after messages have been sent but not read and acted upon.

Analyses of the Proposed System: The architecture design introduces to the generic architecture the Role Gatekeeper to mediate communication between users and the message servers. The Role Gatekeeper implements a proxy pattern [13]. It interacts with the MUA in the same way as a MSA and a MS, and interacts with the MSA and MS in the same way as the MUA. Thus, the Role Gatekeeper enables access to interpersonal messages. The main components of the architecture are the Message User Agent (MUA), the Message Transfer Agent (MTA), the Internet Mail Access Protocol 4 server (IMAP4 server), and the Role Gatekeeper. Figure 2 shows the architectural design framework of a secure role-based messaging system.

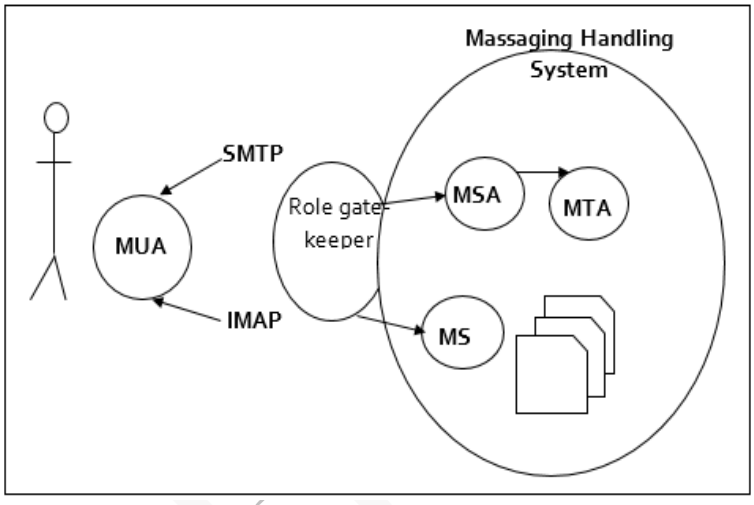



Figure 2: Architecture design framework of a secure role based messaging system

The Role Gatekeeper intercepts messages between the MUA and the MTA server, and communications between the MUA and the IMAP4 server. The communications between the Role Gatekeeper and other components are secure by means of authentication and encryption of messages or links. The Role Gatekeeper is responsible for all security operations regarding roles. The responsibilities of the Role Gatekeeper include imposing role-based access control, managing role mailboxes, and providing security features for role-based messaging.

Design: System design is the process of structuring the system understudy following the specifications of the processing requirements. The design objectives of the study scope around: a framework for a flexible role-based secure messaging service system within an organization, an automatic password generator, and user access code mechanisms that will be sent to the user's mobile phone directly. Others include the design of a two-way message encryption for registered role-based users' mobile phones to ensure privacy and a security system that ensures

confidentiality of the content of mail exchanged within the organization.

For the purposes of development ease, we present a data flow diagram (Figure 3). This diagram is a graphical representation of the flow of data through an information system. It mainly reveals relationships between entities by showing what data comes from where, where it is going to, and where it will be stored.

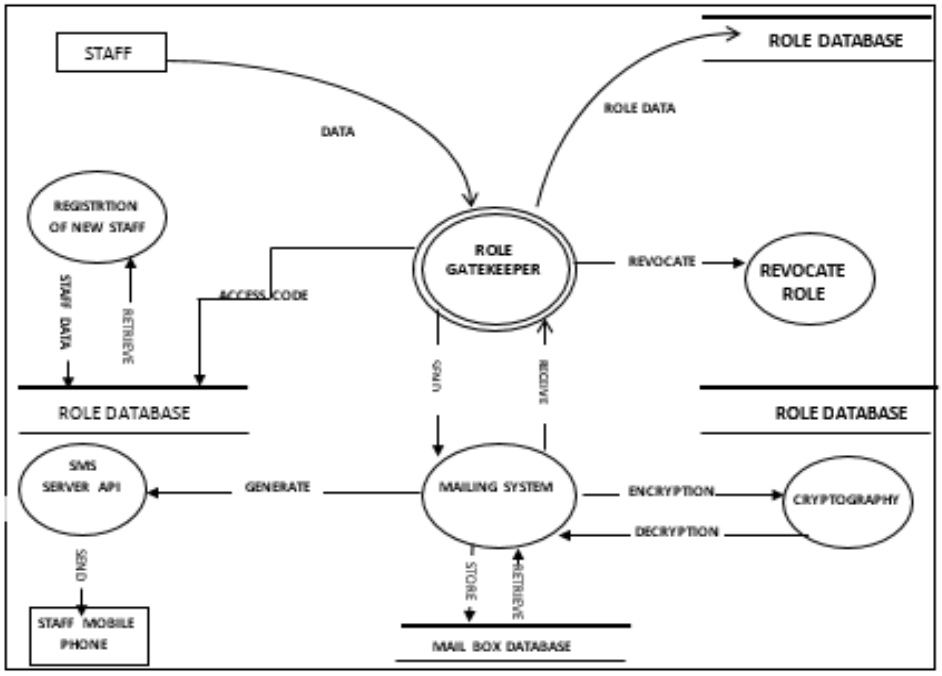



Figure 3: Dataflow Diagram

The high-level model depicts the menus and submenus of the system. They also show the various options/operations or activities which take place within a submenu. This model (Figure 4) is for the proposed system and is thus presented below.

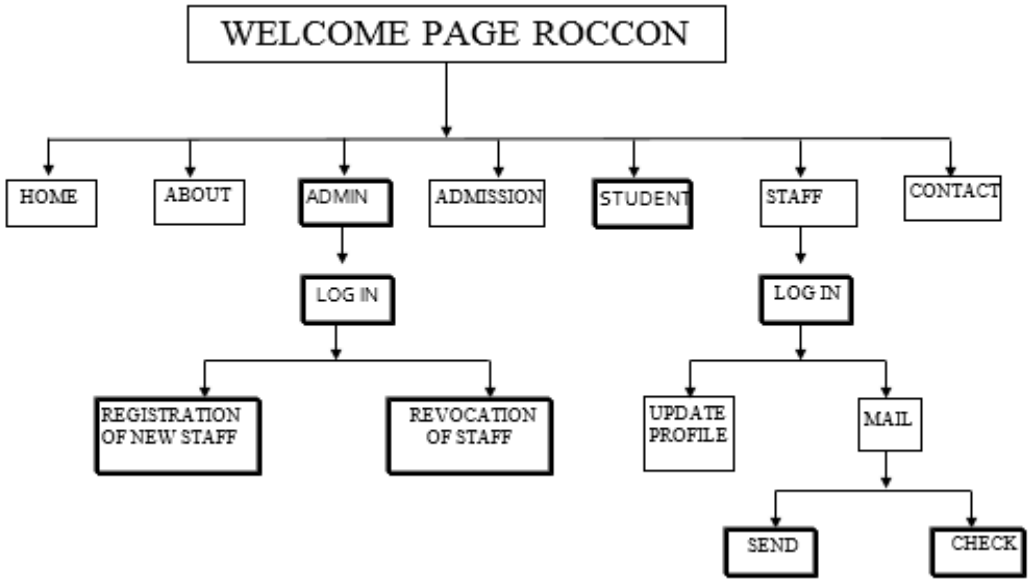



Figure 4: High Level Model (HLM) of the Proposed System

Coding: The choice of Programming language is one of the most important aspects among factors of consideration during program development. This type of core technical and system-oriented software for large and secure environments, like a cooperative messaging system for an organization, would be better implemented using core system oriented languages like C++, Java Script, Visual Basic.net (VB.Net), ASP, ColdFusion, Perl etc. However, we chose PHP for reasons, which include its simplicity, comprehension and flexibility. In addition, PHP is open source, allows appreciable development speed, is portable, compatible with IIS/Apache servers, more stable and less-resource intensive. Also, it supports different databases, such as Postgre SQL, Generic ODBC, MySQL, Oracle, Informix, Sybase, and Solid.

Various forms are to be completed for the various options of services available. The software is designed to be interactive through the keyboard and the mouse. After data has been imputed, the system communicates with the user through the prompt created. The system processes and returns a message to inform the user of a successful operation or otherwise. A sample of various forms is shown below. Figure 5 shows the welcome interface platform.

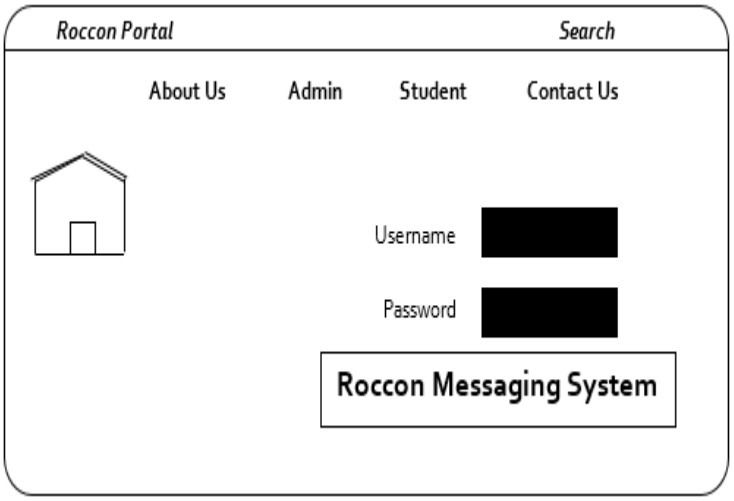



Figure 5: Welcome Interface Platform

System Implementation: this is the development, installation, and testing of system components and the delivery of that system into production. The purpose of this phase is to convert the physical system specifications into working and reliable software and hardware; document the work that has been done; and provide help for current and future users. The technical requirements for hardware and software are as follows. For hardware requirements, this application should be installed on a system running on at least a branded machine or an HP compatible clone that runs on the Microsoft Windows Environment Operating System (Window Vista and above) or any other operating system with at least 1G RAM and a minimum space of 100MB on the hard disk. Software requirements refer to the non-physical part of the computer system that controls the operations of the computer. It is the program that gives instructions needed to efficiently implement the system. These phases include the documentation of the code, algorithms, interfaces, and APIs. Coding is an activity where all the designs created during the previous design phase are programmed using software.

This project is a very technical program. It's system-oriented software for a corporate organization and is better implemented using a scripting language. Because the program must be web-based, an object-oriented language is used-PHP version 5.3.13. PHP (formerly Personal

Home Page, now PHP: Hypertext Preprocessor) is a 32bit operating system that can run on Microsoft Windows. A server-side scripting language designed for web development. It is also a general-purpose programming language which can be deployed on most web servers and as a standalone shell on almost every operating system or platform.

**Conclusion**

Herein, we designed and implemented a framework for developing a role-based secure messaging service which would serve as a roadmap or guide for messaging security designers, implementers, or organizations. Specifically, we have successfully modeled a framework for a flexible role-based secure messaging service system which would not only serve as a roadmap or guide but also a building plan for implementing a secure messaging system in our corporate organization. This research shows that it is viable to use secure role-based messaging systems to provide secure messaging for commercial applications. The significance of the study includes: the frame work of a flexible role-based secure messaging service for a large and structured organization where people often want to send a message to a given role within that organization, rather than to a specific individual. Especially for organizations that highly desire, satisfy and protection of their data and information as well as network infrastructure and resources. The protect protection of messages across domains, and can cater to most of the organizational and commercial security requirements for information confidentiality and some of the commercial requirements for information integrity. This contributes to the design and development related to software engineering issues that help in building the secure role-based messaging model by evolving existing systems and protecting the target systems through special system architect design. The study results are especially useful in large organizations as they will build confidence in the existing messaging system in any organization and put some hackers out of work. Similarly, it will contribute to providing security features based on architecture design, and demonstrate the deliberative architecture design for information confidentiality and privacy.

**References**

[1] Alhogail, A. and Alsabih, A. (2021). Applying machine learning and natural language processing to detect phishing email. Computers & Security, 110. https://doi.org/10.1016/j.cose.2021.102414.

[2] Allman E., Callas J., Delany M., Libbey M., Fenton J., and Thomas M. Domain Keys Identified Mail (DKIM) Signatures. RFC 4871 (Proposed Standard), May 2007. URL http://www.ietf.org/rfc/rfc4871.txt.

[3] American national Standards Institute, Inc. Role-Based Access Control. ANSI INCIT 359-2004., retrieved May 10, 2021.

[4] Chadwick, D. (2003). The X.509 Privilege Management Infrastructure. In Borka Jerman-Blazic, Wolfgang Schneider, and Tomaz Klobucar, editors, proceedings of the NATO Advanced Networking Workshop on Advanced Security Technologies in Networking, Bled, Slovenia, September, 2003.

[5] Chadwick, D., Zhao, G., Otenko, S., Laborde R., and Su, L. (2006), Building a modular authorization infrastructure. In UK e-Science All Hands Meeting 2006, September, 2006.

[6] Chadwick, D., Lunt, G., and Zhao G. (2004). Secure Role based Messaging. In Eighth IFIP TC-6 tc-11 Conference on Communications and Multimedia Security (CMS 2004), Windermere, UK, September, 2004.

[7] CCITT (1988) Recommendations X.400, Message Handling System and Services Overview. Retrieved May 12, 1013.

[8] Crispin, M. (2003). RFC 3501-Internet Message Access Protocol-Version 4revl. Request for Comment, Network Working Group.

[9] Crocker, D. H. (1982). RFC 822-Standard for ARPA Internet Text Message Request for Comment, Network Working Group.

[10] Ekins, M. (1996). RFC 2015-MIME Security with pretty Good Privacy (PGP) Request for Comment, Network Working Group.

[11] Elkins, M., Del-Torto, D., Leveien, R., and Roessler, T. (2001). RFC 3156 _MIME Security with Open PGP Request for Comment, Network Working Group.

[12] Freed, N. and Borenstein, N. (1996). RFC 2045- Multipurpose Internet Mail Extension (MIME) Part One: Format of Internet Message Bodies. Request for Comment, Network Working Group. Retrieved August 18, 2021.

[13] Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesly.

[14] Gellens, R. and Klensin, J. (1998), RFC 2476-Message Submission. Request for Comment, Network Working Group.

[15] Hasselbach, J. (2003). A Practice-Oriented Approach to Security Enhanced Mailing Lists. In The Sixth International Conference on Electronic Commerce Research, Dallas, US.

[16] Hoffman, P. (1999). RFC 2634-Enhanced Security Services for S/MIME. Request for Comment, Network Working Group.

[17] Klensin, J. (2001). RFC 2821-Simple Mail Transfer Protocol. Request for Comment, Network Working Group.

[18] Klensin, J., Freed, N., Rose M., Stefferud, E., and Crocker, D. (1995). RFC 1869-SMTP Service Extension. Requsr for Comment, Network Working Group. Retrieved August 18, 2021.

[19] Levine, J. R. (2004). The Trusted Email Open Standard (TEOS). Internet Engineering Task Force. 1 – 8. https://datatracker.ietf.org/doc/html/draft-levine-epg-teos-00.

[20 ]Lyon, J., and Wong, M. (2004). Internet Draft: Sender ID: Authenticating E-Mail. Internet Draft, Retrieved May 12, 2021.

[21] Lyon, (2005). Purported Responsible Address in E-Mail Messages. Microsoft Corp. https://tools.ietf.org/pdf/draft-lyon-senderid-pra-00.pdf

[22] Manji, K., Hanefeld, J., Vearey, J., Walls, H. and De Gruchy, T. (2021). Using WhatsApp messenger for health systems research: a scoping review of available literature. Health Policy and Planning, 36, 2021, 774–789

[23] Matthew Martin. 2021. SDLC: Phases & Models of Software Development Life Cycle. Guru99. https://www.guru99.com/software-development-life-cycle-tutorial.html#11

[24] Mont, M. C., Bramhall, P., and Harrison, K. (2003). A Flexible Role-based Secure Messaging Service: Exploiting IBE Technology for Privacy in Health Care. In Proceeding of the 14th International Workshop on Database and Expert System Applications. IEEE

[25 ]Myers, J. (1996). RFC 1939-Post Office Protocol-Version 3. Request for Comment, Network Working Group. Retrieved August 18, 2021.

[26] Oppliger, R., Gunther, P., and Christine S. (2000). Using attribute certificates to implement role based authorization and access controls. In S. Teufel K. Bauknecht, editor, Scherheit in Information System (SIS 2000), pp 169-184, Zurich.

[27] Osborn, S. L. (2002). Information flow analysis of an rbac system. In Sacmat '02: Proceedings of the seventh ACM symposium on Access control models and technologies, 2002.

[28] Postel, J. B. (1982). RFC 821-Simple Mail Transfer Protocol. Request for Comment, Network Working Group.

[29] Ramsdell, B. (1999). RFC 2633-S'MIME Version 3 Message Specification. Request for Comment, Network Working Group.

[30] Wolthusen S. D. (2003). A Distributed Multipurpose Mail Guard. In The 2003 IEEE Workshop on Information Assurance, West Point, NY, US.

[31] Wong M. and Schlitt W. Sender Policy Framework (SPF) for Authorizing Use of Domains in E-Mail, Version 1. RFC 4408 (Experimental), April 2006. URL http://www.ietf.org/rfc/rfc4408.txt