

Enhancing mobile agent protection using a hybrid security framework combining pretty good protocol and code obfuscation

Jamal Zraqou¹, Wesam Alkhadour², Mahmoud Baklizi¹, Khalil Omar¹, Hussam Fakhouri³

¹Department of Computer Science, Faculty of Information Technology, University of Petra, Amman, Jordan

²Department of E-Business and Commerce, Faculty of Administrative and Financial Sciences, University of Petra, Amman, Jordan

³Department of Artificial Intelligence and Data Science, Faculty of Information Technology, University of Petra, Amman, Jordan

Article Info

Article history:

Received Jun 13, 2024

Revised Dec 16, 2024

Accepted Jan 14, 2025

Keywords:

Aglets

Code obfuscation

Cryptography

Mobile agent

Pretty good protocol

ABSTRACT

The security of mobile agents, which are autonomous software entities capable of migrating between computers to execute tasks, remains a critical concern in modern information technology. Cybersecurity has been a central component of this technological revolution and continues to be one of the most essential requirements for any software or platform. Despite advances in security measures, protecting mobile agents, particularly those carrying sensitive data, while they transmit over networks remains challenging. This research proposes a novel hybrid security technique, abbreviated as pretty good privacy and code obfuscation framework (PGF), which combines pretty good privacy (PGP) with code obfuscation. PGF is designed specifically to protect mobile agents, focusing on systems like Aglets. The technique aims to safeguard the integrity and confidentiality of the agent's data during transmission. Based on the mobile agent Aglets and the PGF technique, the proposed model enhances security by introducing additional protection layers during agent creation and transmission using PGP and code obfuscation. The comparative analysis demonstrated that PGF outperformed other algorithms in terms of time efficiency and security, effectively handling large data sizes through its hybrid cryptographic approach, which combines asymmetric and symmetric encryption. The model was implemented using the Aglets framework in Java development kit (JDK) and NetBeans and showed high reliability and practicality. However, its current design is tailored to Aglets, and future work could focus on adapting the model to other platforms and optimizing its resource efficiency for constrained environments.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Jamal Zraqou

Department of Computer Science, Faculty of Information Technology, University of Petra

961343 Airport Road, Amman, Jordan

Email: Jamal.Zraqou@uop.edu.jo

1. INTRODUCTION

Although the development of mobile agents has advanced dramatically, the security issue is still a critical problem, especially for the Aglets platform. Many mobile agents like mongers, bullfrogs, and gander. Aglets play an important role in distributed systems because they are self-moving and can perform many tasks in the network. However, some existing agent platforms have been pointed out and supported by recent studies of their weaknesses, especially concerning data integrity and security when in transmission. While general security concepts dominate current literature, there is a less specific and informed dialogue as to how exactly such problems might be encountered in the case of Aglets and what particular limitations they present, namely

their dependency on the Java platform and inadequacy in contemporary cryptography requirements. This research fills the gap by dealing with these specific weaknesses and introducing a proper security model apposite to Aglets with a security model based on a blend of pretty good privacy (PGP) and code obfuscation in protecting these features from new emerging threats.

Aglet is a sophisticated mobile agent platform that facilitates the creation, migration, and execution of mobile agents in distributed computing environments. A mobile agent is a self-contained Java object that has the unique capability to migrate from one host to another within a network. Aglets carry their state and code with them, enabling them to resume execution at their destination seamlessly. The technological solution brings multiple system benefits, including lowered network strain and independent computation management alongside flexible processing in changing operational conditions [1]. Mobile agent technology differs from standard data transfer methods in that it migrates towards data sources for direct processing, followed by result delivery back to clients. The system applies this function to execute tasks more efficiently, particularly within large distributed systems.

Aglets follow a protocol that permits execution suspension on one host, followed by network transmission and resumption on another host with possible behavioral alterations according to encountered environmental factors. The numerous operational possibilities of such systems result in major security risks for the system. Mobile agents encounter untrusted environments after their movements between hosts because these environments may present potential attacks. The system faces security threats from malicious hosts which try to damage agent data and additional malicious agents that can compromise the system integrity [2]. Mobile agents need essential security features to demonstrate effective functionality because their operation happens across diverse distributed systems. Security policies require great difficulty to be uniformly applied over hosts and networks when mobile agents move repeatedly between different administrative domains. The agent must authenticate itself upon reaching any new host before it can access the local resources or services. The exchange between system components for authorizations and permissions, along with passing information, creates numerous security risks. The agent's data transmission between systems becomes vulnerable to man-in-the-middle attacks, while distributed denial of service attacks use host overloads to disable the agent [3].

Hasan *et al.* [4] investigated how cryptographic methods protect mobile agents from MITM attacks. Their findings showed that better encryption protocols applied to agent transmission substantially decrease the chances of interception and tampering. Deshmukh and Chalmeta [5] researched the security challenges of voice user interface (VUI) related to mobile agent security by introducing lightweight encryption protocols, which offer powerful protection alongside minimum resource usage demands. The research demonstrates how security measures compete against performance requirements, specifically when mobile agents run on resource-limited devices.

The research by Zibaeirad *et al.* [6] performed a detailed risk assessment to identify mobile agent system weaknesses against distributed denial of service (DDoS) attacks. Security experts discovered that traditional security practices work well for protecting individual agents, yet the hosts and their entire system need unified protection against attacks. The authors stressed the need to establish security platforms that protect all elements, including agents and their target hosts, since future research requires complete threat management capabilities.

Mobile agent security remains a complex issue even after these recent technological advancements. Security challenges emerge when hostile environments, known as malicious hosts, attack mobile agents by altering their state, redirecting execution, and leaking sensitive data [7]. Security issues become more complex because rogue agents perform hijacking operations and manipulate data while altering code. The encryption techniques that protect agents fall short of defending against every attack type, particularly those that target the agent's code execution process. The effectiveness increases when encryption works together with code obfuscation to protect agent security.

The research introduces a dual-layer security system called pretty good privacy and code obfuscation framework (PGF) that unites PGP encryption with code protection methods to protect agents from cyberattacks. PGP provides encrypted and secure data transmission during all stages of communication to protect agents from MITM attacks, data theft attempts, and sniffing attacks [8]. Code obfuscation works alongside PGP to protect mobile agents by concealing their programming code so that potential attackers cannot analyze the execution logic of mobile agents. The proposed PGF method combines encryption from PGP with code obfuscation to deliver complete security to mobile agents operating in distributed systems, especially within Aglet environments.

PGF provides its main advantage through dual protection of agents' data and execution process. The encryption component safeguards data transmission between hosts, and the obfuscation component shields the agent from being modified at the code level. This method solves multiple limitations found in current methods because they secure either data or code but not simultaneously. Experimental testing of PGF exhibits superior operational performance and better resistance against attack compared to previous approaches. The hybrid

approach protects mobile agents effectively through its performance evaluation of time efficiency while maintaining usability, according to research [9].

The study investigates security threats which affect mobile agents in distributed systems specifically targeting Aglets-based systems. The research will result in the development of a hybrid security method known as PGF. The PGF method protects data transmission security through PGP encryption and code obfuscation to prevent MITM attacks and protect transmitted data from theft and sniffing, as well as shield agent logic from reverse engineering attacks through compromised hosts' IT vulnerabilities. This research adopts these two security techniques to develop an integrated solution that enhances the reliability of mobile agents together with their host systems in distributed networks.

The PGF method presents itself as an effective solution to protect mobile agents operating in distributed networks. By combining PGP and code obfuscation, the proposed technique addresses multiple layers of the security problem, offering enhanced protection for both the agent and the host. Future research should continue to explore the trade-offs between security, performance, and usability, particularly as mobile agent systems become more prevalent in distributed computing environments.

2. BACKGROUND AND RELATED WORKS

2.1. Mobile agent

A mobile agent can be autonomous and sociable but may have to learn tendencies and the most desirable behavior. A mobile agent, more specifically, is a process that gets migrated from one environment to another and works fine in the second environment. The mobile agents themselves are the decision makers of when and where they wish to move. A mobile agent only completes a step by replicating details about it. Mobile agent mobility is when a mobile agent decides to move; the state is written, sent to a new host, and executed. Mobile agents are autonomous software agents that can suspend their execution and resume at another computer connected to the network. The mobility of the code and the state of this agent program are important. Like many other software agents, most programs are self-directed; once they start, they decide where they will go and what they will do. They can receive messages from other sources, but each mobile agent's prerogative is whether to follow them. Such agents may also wish to perform actions such as transferring an entity from one node to another device, jesting, or killing themselves or other agents in the network. Several mobile agent systems with acceptable security degrees were developed, and some of them contain a strong security factor [7].

2.2. Overview of aglets

Aglet specifies an agent system implemented using Java. It travels to nearby distant hosts and deals with events and messages, as shown in Figure 1. Saturated in continual support, an Aglet is defined by a significant description written in Java-lightweight object relocation. The Java aglet builds upon the model of portable code organization made well-known by Java applets. Before an aglet can visit another personal computer (PC) on a local area network (LAN), it must have a host Java program running on that PC. Aglets move from one aglet host to another as they are transferred within an organization. This is achieved by implementing a protection method for every aglet that prevents the occurrence of ulterior Aglets. Hosts deliver aglets via class loaders and class records. The state of an aglet can be recovered from a far aglet host.

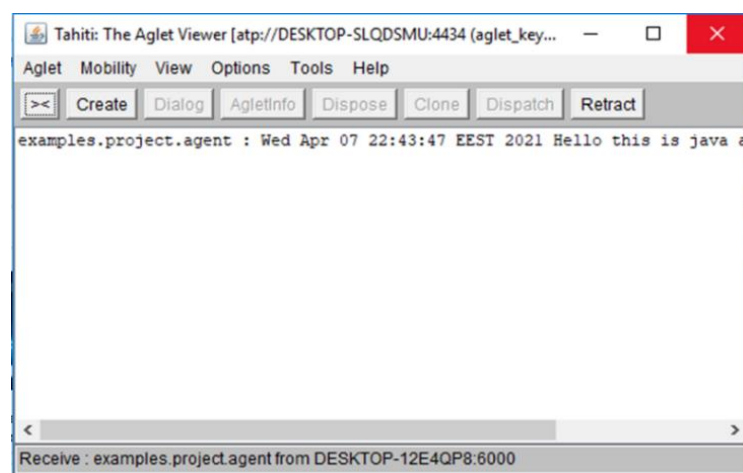


Figure 1. Aglet server

2.3. Obfuscation

Code obfuscation can also be the disassembling of an executable file in such a way that will not be of much use to the programmer anymore but is fully functional. Although changes made by this technique may include modifications to the actual method instructions or the data records linked to them, these changes do not affect the application output. However, there always comes a time when, with so much time and resources, practically any code can be reverse engineered, for instance, on specific Java, Android iOS, NET, Xamarin, C#, VB NET, and F# platforms that are downloadable decompiles that can quickly and rather easily reconstruct initial code out of an executable or a library. Using an automated code has the purpose of making reverse engineering, most of the time, both difficult and costly.

2.4. PGP algorithm

Pretty good privacy, or PGP, stands for an encryption system used for the security of transfer and encoding of files and messages electronically. Due to the mathematical processes involved in encryption, at this point, we will focus only on the basics of encryption. The operational mechanics of PGP encryption, at its most elementary level, are as follows:

PGP then derives a random session key, the key that is used only once understood to be a vastly extensive register of key possibilities difficult to crack. This session key is then encrypted using the recipient's public key. The public key is associated with a specific person's identity to make it possible for anybody to send them an encrypted message.

PGP is a hybrid cryptographic technique that borrows the best from conventional, symmetric, and public key, asymmetric systems. The first operation performed when a user applies PGP to encrypt plaintext is data compression. This process of compression reduces modem transmission time and saves disk space simultaneously, increasing cryptographic security. Many cryptanalytic techniques attempt to break a cipher by exploiting obvious characteristics of plaintext. The extermination of these patterns is achieved by compression, enhancing cryptanalysis resistance by leaps and bounds. It is noteworthy that files that are too small for compression or do not compress well are not included in this process.

Pretty good privacy (PGP) represents an encryption technique used with the aim of secure transfer and secure encryption of sensitive documents as well as electronic communication. Due to the complexity of the mathematical algorithms that are used in encryption, mine and ours will, therefore, only focus on the basic particles for now. The operation of PGP encryption can be delineated at its most elementary level.

PGP starts the action by producing a random session key that is a unique key used only one time; it has millions of variants, making it almost conceivable to guess. This session key is then generated from random data, and the said random data is generated from mouse movements and keystrokes to enhance the randomness of the session key. After this, the session key uses a highly secure and efficient conventional encryption algorithm to encode the plaintext to ciphertext. After the data has been encrypted, the session key is then encrypted with the recipient's public key. For a detailed description of PGP functionality, see the diagram displayed in Figure 2.

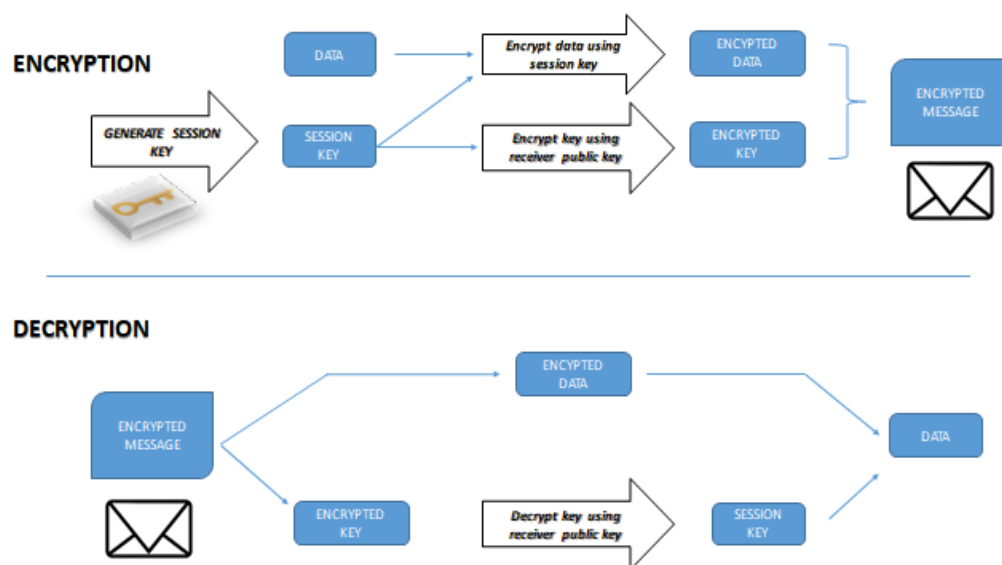


Figure 2. PGP works

3. RELATED WORKS

Advanced mathematical approaches enable the efficient transfer of mobile agents from their starting to ending locations with minimum time requirements in the research. These technical procedures play an essential role in improving both the agent's speed and delay minimization based on findings described in [10]. The research demonstrates how to use the itinerary design pattern alongside the ant colony optimization (ACO) algorithm to determine the ideal patterns through which mobile agents should move. This methodology can achieve both efficiency and dynamic environment adaptability.

The study in [11] explored a rationale advanced encryption standard (AES) in a framework that requires only one key for message encrypting while the sender and receiver agents both possess the same key for decrypting and accessing the data. AES protocol within a mobile agent architecture begins when the server produces an AES key and passes it to the client. The entire message data can be encrypted using the AES key by the client's end and passed back to the server agent, who deciphers the message using the same AES key. This agent then checks the importance of the message contents and goes on to validate the transactional process. Therefore, this mechanism is effective in protecting the transactions over the internet because the AES key length is 128 bits, which would make the task of hackers very difficult.

Similarly, Patel and Patel [12] applied the code obfuscation approach to protect the software industry against illicit reverse engineering. In their study, they provided a detailed discussion of multiple methodologies of obfuscation to strengthen the technical protection of software products. It relies on code transformation applications similar to compiler operations; they used many transformers and evaluated them based on factors such as stealth, reasonable price, and ease of use. They both propose the use of this technique in various fields, including mobile agent systems.

Turab and Zraqou [13] presented a new concept of safe communication whereby an encrypted cipher is embedded into a series of images. For the colored frames of the particular video, the algorithm was applied, and the results were scrutinized based on the text being concealed and eventually restored after the application of the image filtration and video compression.

Peyravi and Latif [14] have described a framework to enhance the security of multi-agent filtering systems by employing the PGP technique for encryption and decryption of messages between filtering agents. Consequently, this approach proved highly beneficial in the redesigning of the agent filtering system to become more efficient. The key rationale for implementing the PGP methodology was based on the belief that having large quantities of information available to the public is not necessarily problematic, as long as this information is spread across a huge variety of databases and sources.

Singh *et al.* [15] highlight the importance of a robust infrastructure in the development and widespread use of agents. In the early stages of research, the scarcity of agent development tools significantly limited the full potential and exploitation of this promising technology. However, with advancements in the field, a wide variety of tools are now available to support the development of robust agent-based infrastructures. Their work provides a comprehensive overview of these tools and offers a detailed comparison of the features they provide, aiding developers in selecting the most appropriate tools for their specific requirements in agent-based system development.

Ahmad [16] highlights that achieving optimal end-to-end quality for video streaming applications requires the integration of advanced networking techniques and content processing strategies. Existing techniques often focus only on the bit stream level, failing to delve into the media content itself. Ahmad emphasizes that video content, which holds vast amounts of valuable information, can be leveraged to predict bit rate and quality more accurately. In a content-aware video streaming framework, this visual content is automatically extracted and utilized to manage video quality dynamically, accommodating both network resource constraints and various quality manipulations.

Amro [17] traces the roots of this concept trace back to the 1960s, when mobile code first emerged in the form of remote job entry systems. Today's mobile agents range from simple distributed objects to highly organized intelligent systems. However, with this evolution comes a rise in critical security issues. Amro discusses the significant work being undertaken to address these challenges, aiming to develop trusted mobile agent systems that can be easily deployed and widely accepted. This paper delivers an extensive analysis of current threats targeting mobile agent systems alongside security targets and protective measures against these dangers. The main goal is to maintain secure mobile agent systems which can find broad acceptance in the market.

Ghosh *et al.* [18] developed an application-level active networking (ALAN) infrastructure that resulted in the creation of the "funnelWeb" implementation. The paper provides an in-depth discussion of the motivation behind this system, its architecture, and its implementation. The authors demonstrate how the ALAN framework enables better web-based coaching and content delivery systems. The authors deployed an active network and active service experimental platform through their infrastructure development. The paper details their active research on application-level routing (ALR), which enhances network-level routing efficiency through application layer integration of active networking principles.

Goswami *et al.* [19] evaluated the performance of three mobile ad hoc networks routing protocols, which included ad hoc on-demand distance vector (AODV), dynamic source routing (DSR), and destination sequenced distance vector (DSDV), using NS-2 for simulations running in large areas. The researchers assessed performance differences of protocols through QoS metrics, which included packet delivery ratio (PDR) and normalized routing overhead together with throughput and jitter when implementing the protocols in MANETs covering expansive areas with fast mobile nodes. The authors used NS-2 simulation tools, AWK data analysis tools, and Xgraph to visualize performance metrics. One-way analysis of variance (ANOVA) tools served to confirm the accuracy of the obtained results. The analysis reaches its conclusion by selecting the routing protocol which demonstrates the best performance based on established conditions.

Günter and Braun [20] explain how a measurement infrastructure needs to exist for new internet protocol (IP) technologies standardized by the Internet Engineering Task Force, including differentiated services and IP security. Internet service providers can deliver better network-level services through these technologies. The authors emphasize the need for a secure platform that provides monitoring capabilities to customers regarding service quality and effective collaboration between service providers. Because of this identified need, a mobile agent-based monitoring infrastructure was developed to measure and manage the new and enhanced IP services.

Mobile agent technology serves as a revolutionary computing paradigm according to Jansen and Karygiannis [21] because software agents can seamlessly continue their execution on another Agent-enabled host after suspending execution on their initial host. The authors document mobile agents as developed from their initial appearance in remote job entry systems during the 1960s into advanced software systems which integrate intelligent components. Advanced computing sophistication has introduced major security threats to the system. The paper delivers an extensive examination of mobile agent technology by presenting its overall structure and discussing built-in security hazards. The paper presents current strategies for vulnerability mitigation along with existing approaches to create secure mobile agent frameworks according to Jansen and Karygiannis.

Kaur and Sra [22] introduced methods to boost wireless sensor network efficiency through their deployment of sensor nodes in unattended areas to collect environmental data. These networks benefit from mobile sinks instead of static sinks because they save energy while increasing network duration. Wireless sensor networks (WSNs) that have both varying designs and battery limitations can successfully be managed through multi-agent systems. The authors develop a genetic algorithm to create concurrent routes for smart mobile data collectors which acquire sensor data throughout restricted timeframes using minimum energy usage. The mobile agents independently modify their route plans to bypass any unforeseen system breakdowns. The researchers applied their proposed algorithms throughout realistic WSN test networks and simulated WSN environments. Empirical results generated through MATLAB version 7.10 confirmed the effectiveness of the approach in various network configurations during multiple experiments and simulations.

Ismail [23] explores the role of security in the widespread acceptance of mobile agents as a paradigm for distributed computing. In large-scale distributed environments, robust protection mechanisms are essential. The paper also outlines a number of protection mechanisms that are specific to mobile agents, with authentication and access control being the responsibility of the mobile agent platform.

The unique aspect of these mechanisms is that each agent can define its own access control policy using an interface definition language (IDL), which promotes modularity and simplifies the programming process. The proposed mechanisms have been evaluated, with performance measurements demonstrating the overhead introduced by these security features. The key advantages of these mechanisms are their transparency to the agents and the ability to port nonsecure applications into a secure environment. Both the mobile agent system and the protection mechanisms have been implemented, with experimental results confirming their feasibility and benefits.

Huhns and Singh [24] explain that ontology functions as a computational model which displays elements from the actual world by using a semantic network structure with concepts or individual objects as nodes and relationships or associations between them as arcs. The networks receive additional elements such as properties, attributes, constraints, functions, and rules to control concept behavior. The formal definition of ontology describes it as a mutual understanding of conceptualization through mechanisms which model domain understanding and create semantic definitions for theoretical frameworks. Ontologies link the names of the ontology links database elements, including classes and relations with human-understandable text definitions that also include specific formal rules for interpretation and usage. Information systems and Internet applications use ontologies to structure keywords and database concepts through semantic relationship capture, which generates domain-specific abstract information spaces for users.

Singh *et al.* [25] investigates mobile agents as self-governing software platforms that learn autonomously and move between systems to execute tasks for users. The research community now place greater emphasis on mobile agents because of their capabilities in mobile computing applications. Mobile

agents serve as tools for mobile computing, which finds applications in network management alongside information management systems. The manuscript delivers a thorough examination of mobile agents, including their definition, characteristics, types, and necessary applications and technical obstacles in mobile systems. The included case study presents mobile agents being used for information retrieval purposes while exhibiting high success rates in this domain.

Research project details about information overload mitigation for computer users are presented by O'Riordan and Sorensen [26]. The project created precise information filtering software which controlled online electronic information dissemination. The system's design integrated statistical information retrieval techniques because they demonstrated scalability and reliability but included artificial intelligence (AI) literature techniques to create the adaptive platform. INFormer represents the final system, which implements intelligent agents for information management through machine learning algorithms alongside adaptation and relevance feedback methods. During processing, the system uses weighted graph representations of documents and graph manipulation algorithms, which improve its performance in information filtering and delivery.

The research by Pandey *et al.* [27] demonstrates how Java enables mobile agent system development by focusing on Java-based mobile aglet agents. Aglets serve as a promising technology that offers streamlined procedures for developing distributed systems through easy design and improved implementation and maintenance features. Systems achieve higher reliability and better fault tolerance when mobile agents operate independently from their source process since they cut down network traffic and minimize latency issues. The paper delivers a technical introduction about aglets, together with a security analysis of mobile agent systems that utilize aglets effectively because they can spread harmful programs. The paper analyzes security concerns along with potential attacks on aglets while presenting a security model which protects these agents. The paper investigates aglet applications and their universal usage potential across multiple situations.

The research paper by Picco [28] establishes mobile agents as a modern abstraction which plays a significant role in distributed application structuring. This paper demonstrates the research domain of mobile agents through a demonstration of their beneficial features. The implementation of mobile agents generates three key advantages through enhanced adaptability and decreased communication burden and enables independent operation between different connected systems. The paper examines the core architecture and supporting technologies of mobile agents together with their design and implementation aspects. The paper acknowledges various outstanding issues that limit broader mobile agent paradigm adoption because of security challenges, interoperability complexities, and performance-related constraints.

The research by Selamat and Omatu [29] explores mobile agent usage for internet data collection from specified sites by optimizing the process duration. To ensure relevant network information collection, the authors suggest determining a threshold value through the total number of mobile agent visits. Genetic algorithms (GA) operate to improve mobile agent routing patterns which minimize query expense without compromising path delays. The proposed routing algorithm achieves better performance across diverse parameters during simulation tests, which establishes an efficient solution for data collection in mobile agent systems.

Shekhar *et al.* [30] analyze the problems that MANETs face because of their use of flooding route request messages to discover routes dynamically. The path discovery process through flooding route request messages produces high latency and large bandwidth consumption, which makes such protocols inefficient for real-time multimedia communication. The authors present MAMR as a mobile agent-aided multicast routing protocol which addresses these issues. MAMR combines intelligent mobile agents with on-demand multicast routing protocols by uniting multicast ad hoc on-demand distance vector (MAODV) and on-demand multicast routing protocol (ODMRP).

4. THE PROPOSED APPROACH

A mobile agent application has been developed that moves between Aglet systems to execute tasks while an encryption layer prevents code analysis yet remains confusing to human readers. The code retains its operational integrity but possesses such complex interpretation that any programmer finds it impossible to understand its content. During the third development phase, pretty good privacy (PGP) encryption joined our application to protect agent information from information breaches while defending against man-in-the-middle (MITM) attacks. Agent data encryption using the PGP algorithm at the fourth stage allows the secure transmission to the host, where the PGP algorithm, for full operational readiness, decrypts the data.

Finally, we measured the time taken over ten cycles and made a comparison with other research papers that utilized agents using different algorithms within mobile agent systems. The time factor stands out as one of the key security parameters when protecting the mobile agents when in transit across the network; the transit of the agents from one node to another ought to be executed in the least amount of time possible. The temporal aspect was also a critical factor during the experiment because it is important to evaluate the effectiveness of

PGP with various amounts of data and to investigate how effectively the technology of session keys secures transmission and communication through the network [31].

4.1. PGP algorithm

First, the OpenPGP library is imported within the Windows operating system environment known as Gpg4win. This library facilitates the generation of two distinct keys: public and private. The public key is also involved in the encryption process and is defined as the ASC (ASCII-armored) file type that can be modified. Retrieving the public key requires using the class *PGPPublicKey* embedded in the library. However, it is necessary to pass the private key to the person who will perform the decryption task afterward. The private key, like the public key, is an ASC file type and, like the public key, can be altered before transmission. To get the private key, it is suggested to use the *PGPSecretKey* class, which is located in the given library [32].

4.2. Code obfuscation

Obfuscation is the deliberate act of making the source code, or other computer code, as difficult to comprehend to a human being as possible. Programmers may write code in such a way as to try to make the purpose of the code difficult to determine, as well as the logic it uses and the values it embodies. The primary motivations for doing this are to protect it from being changed, discourage people from understanding how it works, or make it a game for anyone reading the source code. There are scenarios in which it will be done manually, and there are others in which some tools will be used, which are most commonly used in the industry being automated tools [33].

5. RESULTS ANALYSIS

5.1. Executing mobile agent

The Aglet framework is the development platform for mobile agents, through which programmers create mobile agents. At this point, the system reaches full readiness to proceed to the next step, as shown in Figure 3, while the agent code undergoes obfuscation. Code obfuscation increases the security of mobile agents because it makes the underlying implementation harder to analyze or reverse engineer.

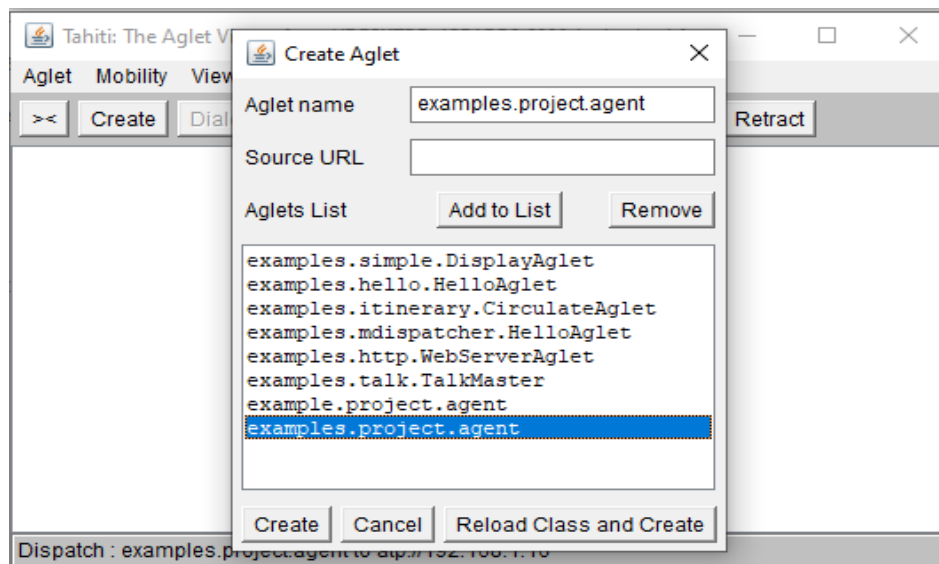


Figure 3. The graphical user interface of the Aglet remote agent management

5.2. Applying code obfuscation

The transformation of variable names to unrelated names so that they did not hold any meaningful information regarding their content or functionality formed the aspect of code obfuscation. Also, we improved the program's readability by adding more and more levels to the program's variable steps so that people who study the code cannot understand what the program does easily. Figure 4 shows another method of data transformation used in the code of the agent-code obfuscation as performed in [34], [35]. This paper promotes

promising future research in e-spam filtering by integrating an information gain processor, a wrapper grey wolf optimizer feature selection algorithm, and a naive Bayes classifier.

```
public String a() {
    String b = "";
    try
    {
        b =
            "Hello this is java agent on AGLET server!!!"
        ;
    } catch
        (Exception e)
    {
        e.printStackTrace();
    }
    return b;
}
```

Figure 4. Code obfuscation

5.3. Encryption

Having invoked the mobile agent with Aglet, the next step is performing any further operations with the NetBeans Platform. Here, we use the PGP algorithm to encrypt the earlier computed session key with the receiver's public key. This documented session key is then used to encode and decode the agent's data content. Figure 5 illustrates how the content was encrypted as envisaged.

```
// add Bouncy JCE Provider, http://bouncycastle.org/latest\_releases.html
Security.addProvider(new org.bouncycastle.jce.provider.BouncyCastleProvider());

// hardcode our private key password **NOT A GOOD IDEA...duh**
String privateKeyPassword = "hongkong";

PGPPublicKey pubKey = null;
// Load public key
try {
    InputStream input;
    input = new FileInputStream("sign-and-encrypt_pub.asc");
    pubKey = readPublicKey(input);
} catch (PGPException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
} catch (IOException a) {
    // TODO Auto-generated catch block
    a.printStackTrace();
}
```

t-encryptanddecrypt (run) ×

examples.project.agent : Wed Apr 28 21:24:47 Hello this is java agent on AGLET server !!!

Successfully read public key:

N2JlZilZf5uu3CHj7KehupJXmv2bEH6fKCVvTuFYwPL3e+0aJRVEmzazQLgBaTV8qxpqv5QjV4QK
Wl2Mma7/IVmISLR0wd0nK4DeBCPIfJp1YUev3xIdOQmZ8wNM8J+9

Figure 5. Applying encryption on the sender Aglet machine

5.4. Decryption

In the process of this stage, the encrypted agent's content was decrypted using the PGP algorithm. This involves the operation of encrypting the session key using the receiver's private key on the destination side. The secret data was decrypted using the session key after the session key was decrypted. As depicted in Figure 6, the decryption was accomplished as desired.

```
PGPSecretKey pgpSec = null;
try {
    InputStream input = new FileInputStream("sign-and-encrypt_priv.asc");

    pgpSec = readSecretKey(input);
} catch (IOException | PGPEException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}

// sign our message
String messageSignature = null;
try {
    messageSignature = signMessageByteArray(message, pgpSec,
        privateKeyPassword.toCharArray());
} catch (NoSuchAlgorithmException | NoSuchProviderException
    | SignatureException | IOException | PGPEException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}
```

Figure 6. Decryption of the destination Aglets machine

6. TIME ANALYSIS

When comparing adaptive algorithms, it is important to take quality and time of execution into consideration. We did 10 independent runs on file sizes of 100, 500, and 1,000 kB presented in Table 1, ensuring we recorded the elapsed time in each experiment. The performance of the PGF approach was then compared with that of ACO, GA, and neural cryptographic algorithm (NCA) with the help of Figures 7-9, respectively. When the data size is kept at 100 kB, as in the experiment's first phase, the average execution time with the PGF is 460.290 milliseconds out of ten cycles.

Table 1. Accumulated results with data sizes 100, 500, and 100 kB

	1000 kB	500 kB	100 kB
GA	3495.98	2058.34	491.32
NCA	2964.98	1697.14	431.26
ACO	2359.49	1447.81	414.33
PGF	2119.29	1566.18	461.29

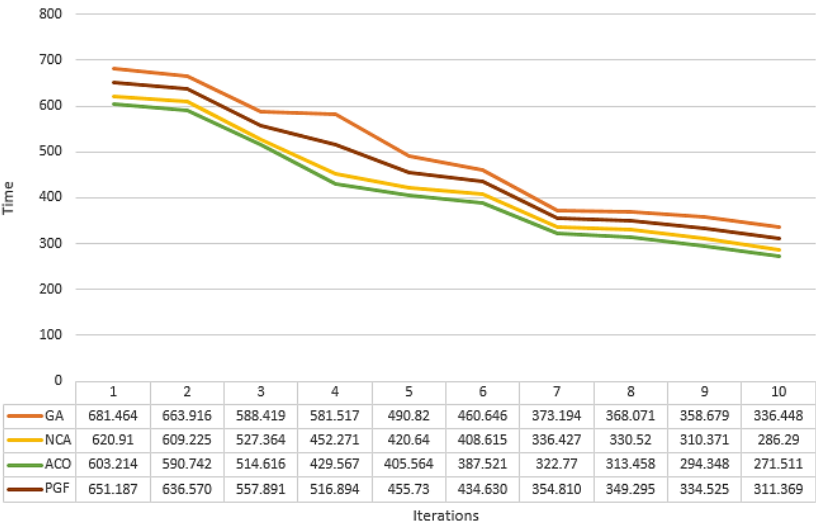


Figure 7. Result at data size of 100 kB

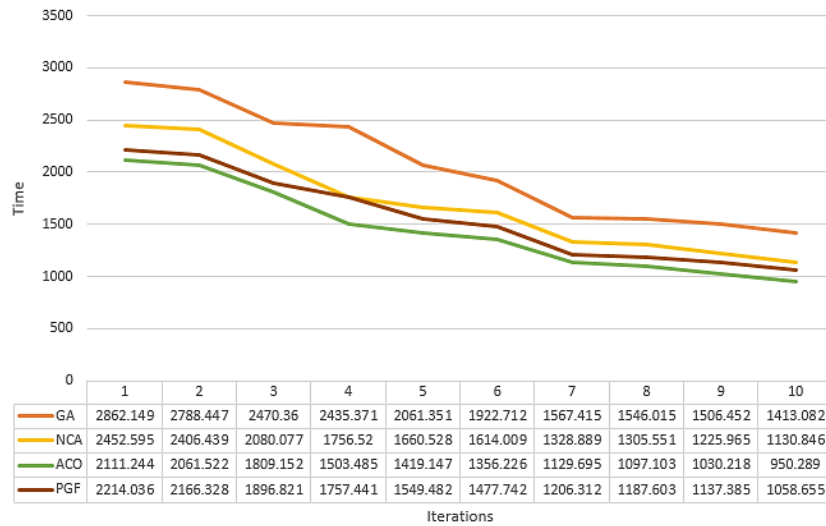


Figure 8. Result comparison 500 kB

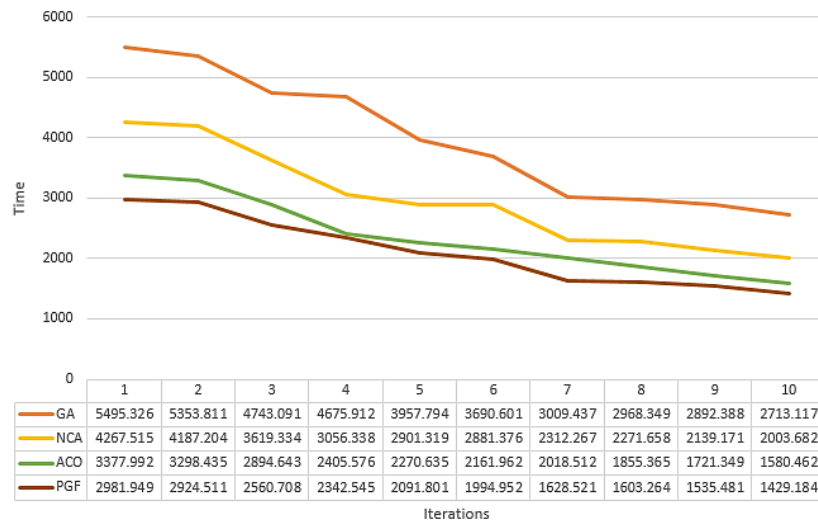


Figure 9. Result with data size of 1000 kB

In the next phase of our experiments, we extended the data size up to 500 kB. At this larger size, the overall execution time of the PGF approach improved and became more acceptable compared to other algorithms tested. It took an execution time of 1565.181 ms, significantly improving their performance. Finally, a 1,000 kB of data size was used, and the 2109.292 ms was counted. Experiments were performed on a PC with Windows 10 64-bit OS, CPU 2.20 GHz, and main memory 16 GB.

7. CONCLUSION

This research paper provides a security and reliability model for Aglet mobile agents by implementing the PGF technique alongside extended network transmission protection in agent creation and distribution. PGP encryption and code obfuscation provided our defense against increasing cyber threats because they represent established security mechanisms. According to our comparison study, the PGF method proves satisfactory against alternative agent systems because of its time-efficient performance. The Aglet agent system achieved better security and performance gains after implementing our method, among other systems that relied on different algorithms during assessment.

PGF can cope with large data sizes because it employs both asymmetric and symmetric approaches to data encryption. It uses both the public and session keys for encryption and decryption, respectively, thus speeding up the time taken for decryption and overall concretization. This use of a session key can be credited

with constituting an evolutionary feature of today's cryptographic models. Last, the proposed model was coded using Aglets in the Java development kit (JDK) while practicing in the NetBeans integrated development environment (IDE).

8. FUTURE WORK

This study suggests a new security model that can improve the reliability of mobile agent Aglets by adding extra security protection layers using the PGF technique during the creation of the agent and the transmission of the agent all over the network. We use PGP encryption and code obfuscation as strong protective measures to mitigate the constant rise in the diversity of attacks. In comparison with the other agent systems where different algorithms were used, the proposed approach showed reasonable performance, especially in terms of time needed for the execution of the task. The PGF method provides more security and utilizes less computational time for large data sets than a conventional system because the PGF uses both asymmetric and symmetric key cryptography. However, public key cryptography is used for encryption. In contrast, a session key is used to implement the decryption, a new era's philosophy of using public and private keys. The model was developed using Aglets under the Java Development Kit - JDK and NetBeans IDE.

ACKNOWLEDGMENTS

The research authors strongly thank the University of Petra for its enduring backing and motivational support during their research period. University resources and facilities played a significant role in concluding this research work. Our appreciation goes to the colleagues and staff members for their valuable contributions during the different phases of the research.

FUNDING INFORMATION

Research funding for this work originated from the University of Petra through its provision of publication expenses. The authors express sincere appreciation for the University of Petra's facilitation of this publication, which was made possible through institutional financial support.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Jamal Zraqou	✓	✓	✓	✓	✓	✓		✓	✓				✓	
Wesam Alkhadour		✓				✓		✓	✓	✓	✓	✓		
Mahmoud Baklizi	✓		✓			✓					✓		✓	
Khalil Omar	✓			✓			✓							✓
Hussam Fakhouri					✓		✓			✓		✓		✓

C : Conceptualization

M : Methodology

So : Software

Va : Validation

Fo : Formal analysis

I : Investigation

R : Resources

D : Data Curation

O : Writing - Original Draft

E : Writing - Review & Editing

Vi : Visualization

Su : Supervision

P : Project administration

Fu : Funding acquisition

CONFLICT OF INTEREST STATEMENT

According to the authors, this research does not involve financial conflicts or personal relationships that might affect its findings.

INFORMED CONSENT

Human subject research was absent from this study due to its non-use of participants or any personal data that needed informed consent since it did not apply to this study.

ETHICAL APPROVAL

When papers talk about using people or animals, authors should make it clear that the research followed all national rules and institutional policies, and it was approved by the authors' institutional review board or a similar committee. The Helsinki Declaration's tenets must guide all investigations involving human subjects. Authors must also identify the committee or review board approving the experiments and provide a statement indicating approval of the research. Incorporate the following (or a similar) statement: The research related to human use has been complied with all the relevant national regulations and institutional policies in accordance with the tenets of the Helsinki Declaration and has been approved by the authors' institutional review board or equivalent committee; or: The research related to animal use has been complied with all the relevant national regulations and institutional policies for the care and use of animals.

ETHICAL APPROVAL

The research project involved no human subjects or animal usage and did not require ethical approval procedures. This research followed the applicable institutional guidelines and ethical protocols. Since the study lacked human or animal participants, it did not need formal ethical approval.

DATA AVAILABILITY

Derived data supporting the findings of this study are available from the corresponding author, JZ, upon request. The libraries that were used in this research are as follows:

- JavaAglets API: <http://aglets.sourceforge.net>
- NetBeans IDE: <https://netbeans.apache.org>
- Aglets in NetBeans: After adding the Aglet library to your project, you can create Aglets that can travel over networks, interact with remote services, and execute tasks.




REFERENCES

- [1] D. B. Lange and M. Oshima, "Mobile agents with Java: the Aglet API," *World Wide Web*, vol. 1, pp. 1–18, 1998.
- [2] D. B. Lange, M. Oshima, G. Karjoth, and K. Kosaka, "Aglets: programming mobile agents in Java," 1997, pp. 253–266, doi: 10.1007/3-540-63343-X_52.
- [3] S. Bijani and D. Robertson, "A review of attacks and security approaches in open multi-agent systems," *Artificial Intelligence Review*, vol. 42, no. 4, pp. 607–636, 2014, doi: 10.1007/s10462-012-9343-1.
- [4] M. K. Hasan *et al.*, "A review on security threats, vulnerabilities, and counter measures of 5G enabled internet-of-medical-things," *IET Communications*, vol. 16, no. 5, pp. 421–432, Mar. 2022, doi: 10.1049/cmu2.12301.
- [5] A. M. Deshmukh and R. Chalmeta, "User experience and usability of voice user interfaces: a systematic literature review," *Information*, vol. 15, no. 9, p. 579, Sep. 2024, doi: 10.3390/info15090579.
- [6] A. Zibaeirad, F. Koleini, S. Bi, T. Hou, and T. Wang, "A comprehensive survey on the security of smart grid: challenges, mitigations, and future research opportunities," *arXiv preprint arXiv:2407.07966*, 2024.
- [7] M. Rezaul Karim, "Security for mobile agents and platforms: securing the code and protecting its integrity," *Journal of Information Technology & Software Engineering*, vol. 08, no. 01, 2017, doi: 10.4172/2165-7866.1000220.
- [8] S. Wang, "PGP encryption software," Thesis, Lapland University of Applied Sciences, 2014.
- [9] D. Emmenegger, "PGF plug-in," Semesterarbeit am Institut für Theoretische Informatik der ETH Zürich, 2001.
- [10] H. Kanaker, N. Abdel Karim, S. A.B. Awwad, N. H.A. Ismail, J. Zraqou, and A. M. F. Al ali, "Trojan horse infection detection in cloud based environment using machine learning," *International Journal of Interactive Mobile Technologies (IJIM)*, vol. 16, no. 24, pp. 81–106, 2022, doi: 10.3991/ijim.v16i24.35763.
- [11] A. Onashoga, A. Akinde, and A. Sodiya, "A strategic review of existing mobile agent- based intrusion detection systems," *Proceedings of the 2009 InSITE Conference*, 2009, doi: 10.28945/3372.
- [12] R. Patel and N. Patel, "A way to protect software secrets from reverse engineering using code obfuscation techniques," in *International Conference On Advance Computing and Creating Entrepreneurs (ICOACCE'10)*, 2014, no. December, pp. 89–92.
- [13] N. M. Turab and J. Zraqou, "A robust approach for applying a secure data transmission," *Journal of Theoretical and Applied Information Technology*, vol. 97, no. 12, 2019.
- [14] F. Peyravi and A. Latif, "Secure multi agent information filtering," *International Journal of Computer Theory and Engineering*, vol. 6, no. 3, pp. 240–246, 2014, doi: 10.7763/IJCTE.2014.V6.869.
- [15] A. Singh, D. Juneja, and A. K. Sharma, "Agent development toolkits," *International Journal of Advancements in Technology*, vol. 2, no. 1, pp. 158–164, 2011.
- [16] A. M. A. Ahmad, "Content-based video streaming approaches and challenges," *Handbook of Research on Mobile Multimedia*, pp. 357–367, 2006, doi: 10.4018/978-1-59140-866-6.ch024.
- [17] B. Amro, "Mobile agent systems, recent security threats and counter measures," *IJCSI International Journal of Computer Science Issues*, vol. 11, no. 2, pp. 146–151, 2014.
- [18] A. Ghosh, M. Fry, and G. MacLarty, "An infrastructure for application level active networking," *Computer Networks*, vol. 36, no. 1, pp. 5–20, 2001, doi: 10.1016/S1389-1286(01)00153-0.
- [19] S. Goswami, S. Joardar, C. B. Das, S. Kar, and D. K. Pal, "Performance analysis of three routing protocols in MANET Using the NS-2 and ANOVA test with varying speed of nodes," *Ad Hoc Networks*, 2017, doi: 10.5772/66521.
- [20] M. Günter and T. Braun, "Internet service monitoring with mobile agents," *IEEE Network*, vol. 16, no. 3, pp. 22–29, 2002, doi: 10.1109/MNET.2002.1002996.
- [21] W. Jansen and T. Karygiannis, "Mobile agent security," Gaithersburg, MD, 1999, doi: 10.6028/NIST.SP.800-19.
- [22] H. Kaur and M. S. Sra, "Optimization of mobile agent using genetic algorithm in wireless sensor network," *International Journal*




- of Computer Applications, vol. 134, no. 2, pp. 31–46, Jan. 2016, doi: 10.5120/ijca2016907831.
- [23] L. Ismail, “A secure mobile agents platform,” *Journal of Communications*, vol. 3, no. 2, Apr. 2008, doi: 10.4304/jcm.3.2.1-12.
- [24] M. N. Huhns and M. P. Singh, “Ontologies for agents,” *IEEE Internet Computing*, vol. 1, no. 6, pp. 81–83, 1997, doi: 10.1109/4236.643942.
- [25] Y. Singh, K. Gulati, and S. Niranjana, “Dimensions and issues of mobile agent technology,” *arXiv preprint arXiv:1210.4644*, 2012.
- [26] A. O’Riordan and H. Sorensen, “An intelligent agent for high-precision text filtering,” in *Proceedings of the fourth international conference on Information and knowledge management - CIKM ’95*, 1995, pp. 205–211, doi: 10.1145/221270.221569.
- [27] R. Pandey, N. Sharma, and R. Rathore, “Aglets (a java based mobile agent) and its security issue,” *International journal of emerging trends & technology in computer science (IJETTCS)*, vol. 2, no. 4, pp. 107–114, 2013.
- [28] G. Pietro Picco, “Mobile agents: an introduction,” *Microprocessors and Microsystems*, vol. 25, no. 2, pp. 65–74, Apr. 2001, doi: 10.1016/S0141-9331(01)00099-0.
- [29] A. Selamat and S. Omatu, “Analysis on route selection by mobile agents using genetic algorithm,” in *SICE 2003 Annual Conference (IEEE Cat. No.03TH8734)*, 2003, vol. 2, pp. 2088–2093.
- [30] H. M. P. Shekhar, M. A. A. Kumar, and K. S. Ramanatha, “Mobile agents aided multicast routing in mobile ad hoc networks,” in *The 7th International Conference on Advanced Communication Technology, 2005, ICACT 2005.*, 2005, pp. 765–770, doi: 10.1109/ICACT.2005.246063.
- [31] H. N. Fakhouri, A. S. Al-Shamayleh, A. Ishtaiwi, S. N. Makhadmeh, S. N. Fakhouri, and F. Hamad, “Hybrid four vector intelligent metaheuristic with differential evolution for structural single-objective engineering optimization,” *Algorithms*, vol. 17, no. 9, 2024, doi: 10.3390/a17090417.
- [32] A. Abu Qaadon, F. Hamad, and H. Fakhouri, “Facilitating digital accessibility for students with disabilities into information services at Jordanian academic libraries,” *Library Management*, 2024, doi: 10.1108/LM-01-2024-0001.
- [33] A. k. Al Hwaitat and H. N. Fakhouri, “Hybrid artificial protozoa-based JADE for attack detection,” *Applied Sciences (Switzerland)*, vol. 14, no. 18, 2024, doi: 10.3390/app14188280.
- [34] F. Hamad, H. N. Fakhouri, F. Alzghoul, and J. Zraqou, “Development and design of object avoider robot and object, path follower robot based on artificial intelligence,” *Arabian Journal for Science and Engineering*, Jul. 2024, doi: 10.1007/s13369-024-09365-z.
- [35] J. Zraqou, A. H. Al-Helali, W. Maqableh, H. Fakhouri, and W. Alkhadour, “Robust email spam filtering using a hybrid of grey wolf optimizer and Naive Bayes classifier,” *Cybernetics and Information Technologies*, vol. 23, no. 4, pp. 79–90, Nov. 2023, doi: 10.2478/cait-2023-0037.

BIOGRAPHIES OF AUTHORS






Jamal Zraqou    is an associate professor at the Department of Computer Science/Virtual and Augmented Reality, University of Petra, Jordan, where he has been a faculty member since 2022. From 2018-2029, he was also the dean faculty of IT at IU. His research interests include computer vision, virtual and augmented reality, internet of things (IoT), cyber security, and image processing. He can be contacted at email: Jamal.Zraqou@uop.edu.jo.






Wesam Alkhadour    holds a Ph.D. in computer science from the University of Bradford, United Kingdom. She is currently an assistant professor at the University of Petra, Jordan. She can be contacted at email: wissam.alkhadour@uop.edu.jo.






Mahmoud Baklizi    is an associate professor at the University of Petra in Jordan. He received his Ph.D. degree in network security from Sains Malaysia in 2015. He is a member of the academic staff of the Department of Computer Science. His research interests include IoT, cyber security, machine learning, and spam filtering. He can be contacted at email: mbaklizi@uop.edu.jo.



Khalil Omar    is an assistant professor at the University of Petra in Jordan. He received his Ph.D. degree in computer science from the University of Oldenburg, Germany, in 2018. His academic focus is on distance learning and e-learning in combination with software engineering, mobile applications, usability design, adaptive interfaces, and ERP systems. Users can reach him through his email address: komar@uop.edu.jo.



Hussam Fakhouri    serves as an assistant professor at the University of Petra in Jordan. Dr. Fakhouri obtained his Ph.D. in computer science from the University of Jordan. He has extensive research experience. His academic work consists of about 40 research papers that appear in recognized Scopus and Web of Science ISI-indexed journals. The reader can reach him through email at Hussam.Fakhouri@uop.edu.jo.