Modern Mailbomb Attacks: From Attack Mechanisms to Mitigation Techniques

Nabil Islam (w3nabil@gmail.com)

Disclaimer: *This manuscript is a preprint and has not been peer-reviewed. It is intended for early distribution of research findings. Certain sections contain extended explanations for clarity in this version; these may be revised, merged, or shortened in the final journal publication. Readers are encouraged to interpret the findings responsibly, in line with ethical guidelines. Feedback is welcome to improve the work.*

ABSTRACT

This study sought to explore the evolving cybersecurity threat of email denial-of-service, commonly known as Mailbomb attacks, that may affect individuals and organisations. With increasing reliance on email for critical communication, the impact of these attacks has become more severe. As found by several prior incidents, these attacks have skyrocketed and become harder to detect. To our best knowledge, existing studies rarely combine analysis of modern attack scripts, AI-based evasion, and anti-spam filter bypassing, leaving a critical gap that this study addresses. We evaluate modern Mailbomb attacks through controlled simulations using Python scripts and their ability to evade modern anti-spam filters, highlighting weaknesses in current detection mechanisms. The findings from this study may contribute to the design of resilient detection mechanisms against evolving threats.

1. INTRODUCTION

Even after the invention of various communication media such as telephones, postal mail, social media, etc., email remains the most preferred and reliable medium in educational, business and governance communication. Email is widely used to share confidential documents. However, it is increasingly being exploited in denial-of-service (DoS) attacks intended for various objectives.

A Switzerland-based secure email provider, Xorlab, published a cybersecurity report about email floods in late 2024 [1]. The investigation says that attackers sent over 48,000 emails in 24 distinct attack waves. The attack primarily affected 17 individuals. Later, on 21 January 2025, InfoSecurity Magazine published an article stating that a Russian team dropped a Mailbomb attack. Each victim was bombarded with large volumes of spam emails, potentially receiving up to 3,000 in less than an hour [19]. Such incidents indicate that Mailbomb attacks are still a serious threat, and modern anti-spam filters fail against them. Existing studies largely focus on traditional spam or generic email-based DoS, leaving a critical gap in understanding how modern Mailbomb attacks exploit weaknesses in current defences.

This paper makes three key contributions**.** First, it systematically analyses the operational characteristics of modern Mailbomb scripts and contrasts them with traditional spam tools. Second, it benchmarks their ability to bypass machine-learning-based anti-spam filters. Finally, it proposes targeted mitigation strategies designed to address the unique evasion techniques used by these Mailbomb attacks. Together, these contributions provide both practical insights for email providers and a foundation for future research on defending against evolving Mailbomb threats.

2. BACKGROUND

Denial-of-Service (DoS) has been a serious threat to modern servers [18]. The primary goal of a DoS attack is to overwhelm a target server and make it unavailable for users. The attackers usually send a large volume of traffic with various payloads to disrupt any targeted server. The majority of the servers are currently capable of defending themselves against such DoS attacks.

Mailbomb attacks or Email DoS is a variation of Denial-of-Service (DoS). Instead of overwhelming the network layer, the attacker floods an inbox with large volumes of spam or malicious emails. The aim is not merely disruption but resource exhaustion, for example, filling mailbox quotas, overloading the user’s email client, and distracting from security emails. Modern email providers rely on machine learning models to filter spam [2], but attackers continue to evolve techniques that evade these defences. Victims may not even

realise they are under attack until their mailbox storage is consumed or legitimate messages become inaccessible.

Earlier email spam attacks relied on static content. For instance, a script configured to send 100 spam messages will often reuse the same subject line, header, and body content, making detection easier. Many such scripts are openly available in repositories like Exploit-DB and Metasploit. On the other hand, modern Mailbomb scripts randomise headers, subject lines, body text, origin IPs, and even origin devices, making them harder to detect. They can also incorporate multi-threading and proxy routing, further expanding their effectiveness.

Table 1 compares the features of traditional spam scripts and modern Mailbomb scripts. The latter demonstrate far greater sophistication, speed, and evasion potential, underscoring the limitations of existing anti-spam defences.

Feature	Traditional Spam Script	Modern Mailbomb Script
Account usage	Single	Multiple
Supports Multiple Threads	No	Yes
Combine with proxy routing	No	Yes
Dynamic subject lines, Body text, attachments, and header information	No	Yes
Capability to use AI Contents	No	Yes
Time takes to send 100 emails	~100 seconds	~2 seconds
Required technical proficiency	Beginner	Moderate to Advanced
Script Availability	Public (open-source)	Private (Custom-built)
Attack Intent	Phishing	Resource Exhaustion and/or distraction
Evasion against ML filters	Very Low	High

Table 1: Comparison between Traditional spam script and Modern Mailbomb script

3. RELATED WORK

Email spam and Mailbomb attacks have long been studied as persistent cybersecurity challenges. Early anti-spam research largely relied on rule-based and signature-based methods, which were soon defeated by simple obfuscation techniques. With the rise of machine learning, filters began to incorporate statistical and semantic features such as subject lines, sender reputation, message bodies, and header metadata [3][14][17]. Despite these advances, distinguishing between legitimate and malicious content remains difficult, particularly when adversaries deliberately mimic human-like writing styles [15].

Open-source exploit repositories such as Exploit-DB and Metasploit include basic mail flooding scripts, typically designed to send large volumes of identical messages from a single source [5]. While effective against unprotected systems, these approaches are easily flagged by modern spam filters. Recent developments in attacker tooling, however, show a shift toward customisable scripts that leverage techniques such as randomised content generation, distributed SMTP accounts, proxy rotation, and spoofed headers to avoid detection [6].

Emerging studies indicate that even advanced AI-driven anti-spam systems can be bypassed by adaptive, script-based attacks [7][16]. Existing literature on email-based denial-of-service (DoS) attacks has primarily focused on large-scale floods or traditional resource exhaustion. Far less attention has been paid to how Python multi-threaded, AI-generated content, and obfuscation-driven modern Mailbomb attacks can exploit weaknesses in quota enforcement and filtering models.

This research builds on prior findings by systematically evaluating Mailbomb attacks under controlled conditions, demonstrating their effectiveness against modern ML-based filters, and highlighting the specific risks of quota exhaustion and inbox flooding. Unlike earlier works, our study benchmarks a range of obfuscation techniques, including AI-generated content across multiple attack waves and mechanisms, providing empirical evidence of how attackers can bypass production-like defences and offering novel insights into targeted detection and mitigation strategies.

4. THREAT MODEL

This section examines a threat model involving an attacker proficient in the Python programming language and the Kali Linux operating system’s offensive security tools, such

as Tor and proxychains-ng, who does not wish to rely on common Metasploit exploits to launch an email attack. The attacker intends to use a modern Mailbomb script for the attack.

4.1. Attacker Capabilities

The attacker is assumed to know about the Simple Mail Transfer Protocol (SMTP), the Multipurpose Internet Mail Extensions protocols (MIME), and can utilise a script with proxychains-ng and Tor libraries to evade the IP-based blacklisting. They should have the ability to write a Python script using smtplib, openai (generate professional, human-toned content like subject lines, body text, attachment names, etc., using AI), and threading libraries. Generally, they need to have access to multiple (50 for a smaller-scale attack, 5000 for a larger-scale attack) email login credentials to assist the script in breaking through the account-based rate limitation. The attacker's system specification may vary depending on the threads being used during an attack.

4.2. Target Environment

The targeted business/user is assumed to be using a Postfix, Exim, or corporate email server, protected by modern anti-spam and various rate-limiting mechanisms, and hosting a dedicated mail inbox for each user with a specific quota limit. The targeted user is expected to have beginner to moderate technical skills. Their system is not a matter of concern for the attacker.

4.3. Attack Objective

The primary objective is denial of service by exhausting the mailbox storage quota, overloading the user's interface/webpage, disrupting access to legitimate messages, dispatching malicious payloads hidden inside files (for stealing data/monitoring from a computer) and/or links (for stealing browser cookies), diverting attention during a ransomware/malware drop, data breach, and other cyberattacks.

Attackers may have other objectives rather than the one described previously. For instance, in 2021, PUBG MOBILE (a game developed by Tencent Games) faced an email flood attack from the activists of Bangladesh due to some internal problems with the community manager of South Asia, forcing them to assign a dedicated community manager for their game community to stop the email flood.

5. ATTACK MECHANISM

The attack mechanism of a modern Mailbomb attack relies on both the workflow and anti-spam avoidance techniques. Figure 1 illustrates the entire workflow process. However, anti-spam avoidance techniques are evolving every day.

5.1. *Workflow*

The Mailbomb attack follows a sequential workflow:

5.1.1. Target Identification:

The attacker obtains a valid email address, often harvested from public sources such as Facebook, websites, and LinkedIn.

5.1.2. Script Preparation:

A script is prepared using Python, typically utilising libraries such as

- smtplib [8]: to send emails via SMTP
- openai [9] (Novel): to generate content
- threading [10]: to enable the concurrent sending of hundreds of emails.

5.1.3. Server observation (Novel):

Sometimes, attackers send 5 to 10 emails using a static IP address to determine whether advanced tools are necessary or if a basic script may cause a denial-of-service. They usually monitor the behaviour of anti-spam filters by seeing if the IP address is being blacklisted and/or if the rate limitation hinders the email sending process.

5.1.4. Bypassing the anti-spam:

If the attacker’s observation raises the flag, they proceed with the advanced avoidance techniques described in Section 5.2 to bypass detection.

5.1.5. Mass Dispatch Execution:

Emails are dispatched in parallel threads to overwhelm the inbox with various email accounts. The sending frequency, such as the time interval between sending emails, and the batch size, are configurable within the script.

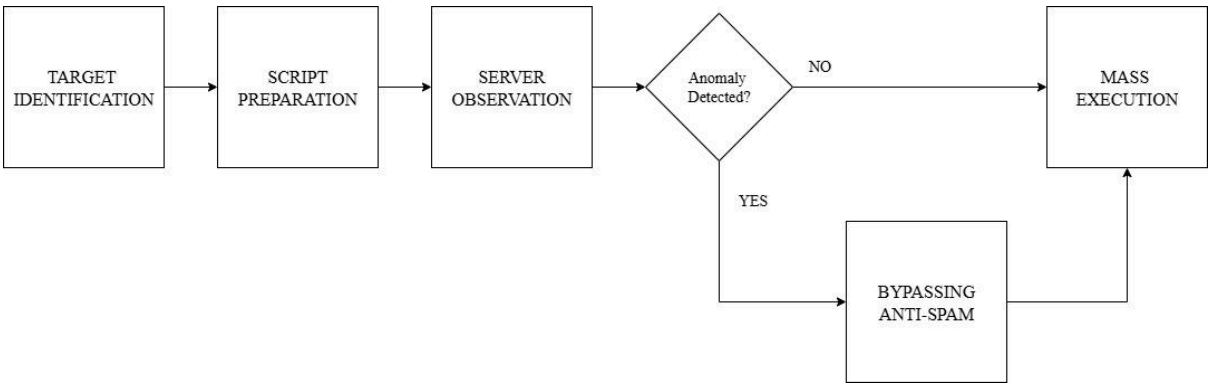


Figure 1: Workflow of a Mailbomb attack.

5.2. Advanced Spam-Filter Avoidance Techniques

To avoid detection by advanced spam filter systems, attackers use:

5.2.1. Legitimate SMTP Headers

Attackers typically observe the SMTP headers of common email providers (e.g., Gmail, Outlook, and Proton Mail) and fill in the necessary headers to impersonate a legitimate sender.

5.2.2. Varied & Realistic AI Content

Using different email body text/subject lines allows anyone to bypass the fingerprinting. Attackers use realistic subject lines (e.g., "Scholarship Request," "Project Collaboration," "Research Submission," "Asking for Reference Letter"), attachments with convincing filenames (e.g., invoice5685_ExampleCakeShop.pdf, John_resume_academic.docx, UGCS_NABIL_RL.pdf), and sender names (e.g., currently enrolled students or teachers, country-based names) that were harvested from the website or social media. To generate such content, attackers use AI prompts to save time.

5.2.3. Low Sending Volume Per IP

The use of proxy chains or free proxies enables attackers to maintain a single IP address that is used to send multiple emails, thereby allowing them to bypass IP-based spam detection.

5.2.4. Standard Email Username Format (Novel)

To make the attack legitimate, the attacker uses a standard email username format, which allows them to bypass ML-based anti-spam filters. Usually, ML-based anti-spam filters block email addresses if they contain random usernames like asdfghj, nabil54535, budapest6357, etc. In contrast, if the email username follows a standard format, these anti-spam filters treat it as a genuine email address and do not place it in the spam folder. Table 2 shows the standard email address format with an example.

Format	Example
[firstname][lastname]@[domain].[tld]	nabilislam@w3nabil.com
[firstname].[lastname initial]@[domain].[tld]	nabil.islam@w3nabil.com
[firstname initial].[lastname]@[domain].[tld]	n.islam@w3nabil.com
[firstname]@[domain].[tld]	nabil@w3nabil.com
[lastname].[firstname]@[domain].[tld]	islam.nabil@w3nabil.com
[firstname]@[subdomain].[domain].[tld]	nabil@mail.w3nabil.com
[prefix][firstname]@[domain].[tld]	drnabil@w3nabil.com

Table 2: A few professional email addresses and username formatting ways which does not trigger ML-based anti-spam flags.

5.2.5. Target email with aliases to reduce the flooding probability (Novel)

Some advanced ML-based anti-spam filters detect the spam attack depending on the volume of emails being received in a period of time to each destination. Around 4 open-source anti-spam filters were investigated. These anti-spam mechanisms only measure the volume received by the destination, and each alias is treated as a different destination. This prevents anti-spam from triggering if aliases are being targeted.

5.2.6. Reputed Email Account (Novel)

Usually, accounts created and used for a certain period are less likely to be flagged as spam accounts. Since they may already have an established sending history and a higher reputation score with email servers, attackers often use such aged accounts to bypass reputation-based

filters compared to newly registered accounts. This makes reputable accounts a valuable resource in Mailbomb attacks, allowing attackers to reduce the likelihood of immediate blocking.

5.2.7. Proxy and Relay Obfuscation

To bypass IP-based rate limiting, the script may integrate with Proxychains4 [11] to route traffic through SOCKS5 proxies and/or Tor (an open-source overlay network for enabling anonymous communication) [12] or OpenVPN (an open-source Virtual Private Network protocol) [13].

6. CASE STUDY AND EXPERIMENTAL EVALUATION

A total of two controlled tests were done to investigate the real-world effects of Mailbomb attacks. In Test 1, a small-scale attack was conducted to highlight their effectiveness against existing anti-spam filters. In test 2, a quota exhaustion attack was performed to demonstrate the weakness of modern ML-based anti-spam filters against a large-scale modern Mailbomb attack.

6.1. Experimental Setup

The target environment consisted of a Kali Linux virtual machine (Ryzen 5 4600G host CPU, 2 vCPUs allocated, 16 GB RAM) running a local Postfix SMTP server with a mailbox quota of 15000 MB to replicate a real production-level email server.
To ensure consistency and measurement accuracy, mailbox quotas were reset after each test/wave, providing a clean baseline for every simulation.

6.2. Experimental Configuration

The programming language Python was used for all test cases. The script followed the flowchart outlined in Figure 2. The attacking email accounts followed the format mentioned in TABLE 2. We have used APPENDIX C to generate the humanised subject line, APPENDIX B to generate the sender name, and APPENDIX A for the body text. Each test's other configurations are listed below:

6.2.1. Test 1 – Effectiveness of Mailbomb Attack Configuration

Test 1 consisted of five waves to evaluate the effectiveness of various obfuscation techniques. The model had access to 100 legitimate SMTP accounts from various email providers. Each account was handled by a separate thread, allowing multiple threads to run concurrently. We sent 100 emails in each wave, as fewer emails per wave allow precise observation of obfuscation effectiveness without overloading the system. Different techniques that we have used across the waves are summarised in Table 3.

Wave No.	Obfuscation Technique					
	Reputed Email Account	Usage of Aliases	Spoofed Header	Randomly AI-Generated Content	Legitimate AI-generated content	Multiple Proxy/IP
Wave 1	Y	Y	-	Y	-	-
Wave 2	-	Y	Y	Y	-	-
Wave 3	Y	Y	Y	-	Y	-
Wave 4	Y	Y	-	Y	-	Y
Wave 5	Y	Y	-	-	Y	Y

Table 3: Specific obfuscation techniques used across waves 1 to 5. 'Y' indicates the following technique was used, and '-' indicates the following technique was absent.

6.2.2. Test 2 – Quota Exhaustion Attack Configuration

The purpose of Test 2 was to measure the time required to fill out the quota. The script had access to 1000 valid accounts from different email providers, including Gmail, Outlook, iCloud, Yahoo, and ProtonMail, which simulates an actual Mailbomb attack. We have conducted the test in two waves. Initially, the anti-spam filters were absent to demonstrate the effects of their absence. However, we have used an advanced machine learning-based anti-spam filter and a random interval to avoid regular interval-based detection after every email being sent in wave 2. As Wave 5 of Test 1 showed great effectiveness (see Figure 3), the same obfuscation technique (see Table 3) was used across the entire Test 2.

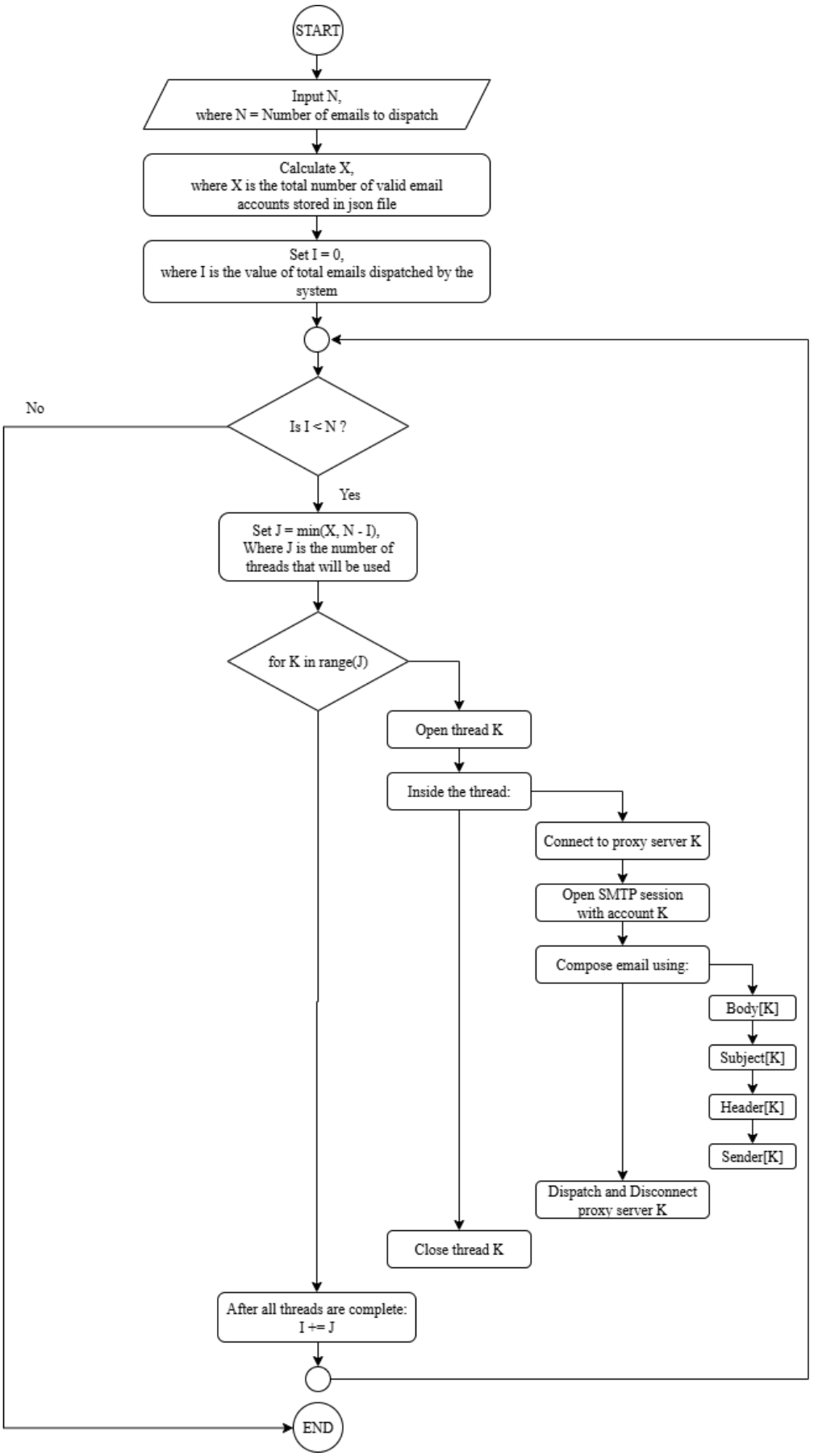


Figure 2: Flowchart of the modern multithreaded Mailbomb script used in Tests 1 (Wave 5) & 2.

6.3. Experimental Results

Figure 3 illustrates the effectiveness of filtering mechanisms across five attack waves. In Wave 1, delivery was mixed, with emails sent to the inbox, spam folder, and blocked by the server. Wave 2 showed stronger blocking, though a portion still reached the inbox. Wave 3 produced mixed outcomes, while Wave 4 highlighted a filter weakness, with the majority of emails landing directly in the inbox. Wave 5 reflected a complete failure, with 100% of emails delivered to the inbox.

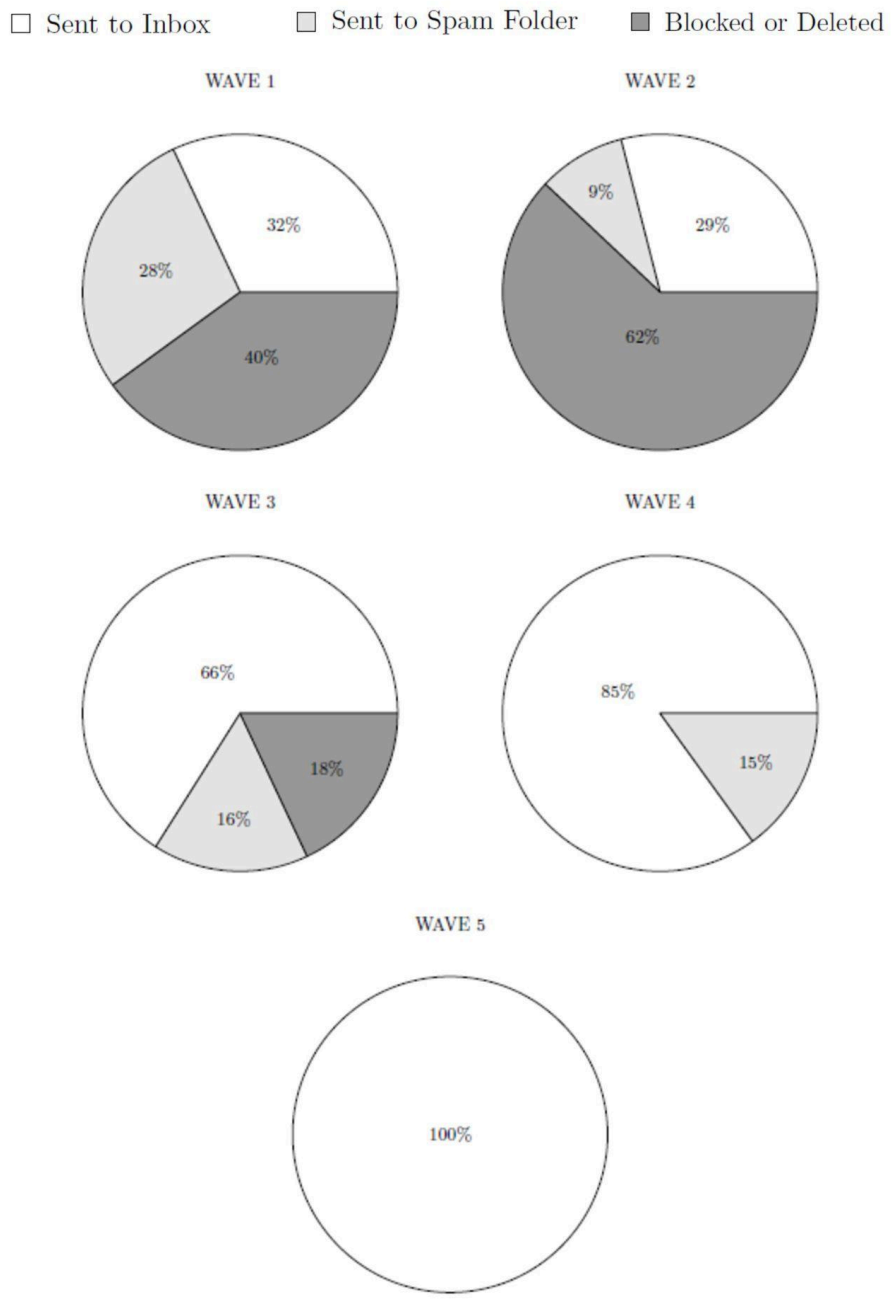


Figure 3: Results of Test

Figure 4 illustrates the impact of quota-filling attacks on server quota usage under two conditions. **Wave 1**, without anti-spam filters, shows a rapid increase in quota usage, reaching 100% within 4 minutes and 6 seconds, indicating a Mailbomb attack may fill the 15GB quota storage faster than expected if any anti-spam filters are not being used. In contrast, **Wave 2**, with anti-spam filters enabled, exhibits a slower rise in quota consumption, demonstrating that a stealth attack may fill the quota within 10 hours.

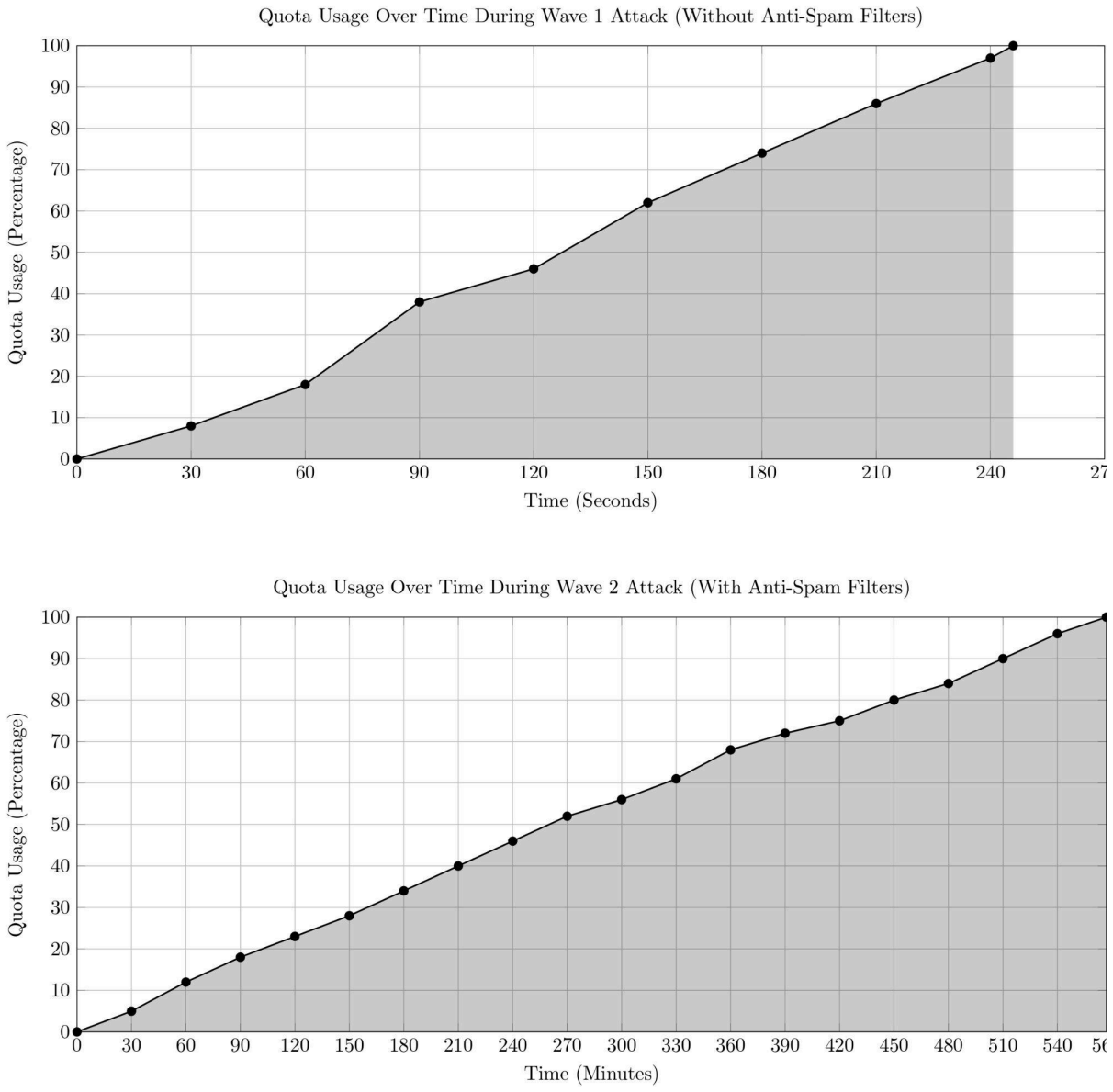


Figure 4: Results of Test 2

6.4. Experimental Discussion

The results demonstrate that a moderately skilled attacker can execute both quota-based and resource-based denial-of-service (DoS) attacks using a simple script. The usage of AI-generated legitimate content and proxies has proven to be highly effective in evading spam filters (see Figure 3). This reveals significant weaknesses in the existing email filtering systems. If not properly configured, this could lead to legitimate messages being lost, which would disrupt communications for both users and businesses. Moreover, such emails may mislead individuals into opening the attachment, leading to remote code execution, session hijacking and malware attacks.

Furthermore, the quota consumption curve (see Figure 4) illustrates that modern Mailbomb attacks can exhaust resources of an inbox within a few hours, without necessarily triggering the rate limits of the standard anti-spam mechanisms. These attacks may allow attackers to bypass the server's incoming email limits, creating opportunities for password attacks. This reveals a significant vulnerability: *anti-spam systems that monitor (i) incoming single-IP, (ii) single-sender accounts, (iii) repetitive-content activity, and (iv) treat aliases differently may fail against attacks spread across multiple senders, IP addresses, and AI-generated content.*

6.5. Experimental Limitations

Due to the absence of official approvals, no tests were performed in production servers such as Gmail, Outlook, or other commercial mail providers. All tests were conducted in a restricted virtual environment. Although an advanced anti-spam was used across the case study, it did not replicate production-level defence mechanisms. While the study demonstrated the viability of an actual attack, real-world impact may vary.

6.6. Experimental Safety Considerations

When performing a large-scale Mailbomb attack using multi-threading, the system may experience CPU overload or overheating. Proper hardware monitoring and thread limitation are recommended to prevent system damage during testing. Additionally, care should be taken to avoid misconfigurations that might unintentionally forward test traffic to production servers (see Section 9).

7. DETECTION STRATEGIES

The combination of behavioural heuristics and log-level analysis can be used at the email server to detect Mailbomb attacks. These tactics aim to identify inappropriate patterns of behaviour that deviate from acceptable email usage.

7.1. Log Analysis

Server logs provide a valuable first line of defence against flooding attacks. Indicators such as similar payload sizes, incomplete metadata failing SPF, DKIM, or DMARC validation, unusually high email frequency within short periods, perfectly regular sending intervals, and the use of IP addresses associated with known VPNs or blacklists can reveal abnormal activity. Repeated attachments or identical hash values in message fingerprints across multiple emails are further signs of automated attacks. Automated log correlation systems can aggregate these signals to flag anomalies and issue early alerts or apply rate-limiting to offending accounts. However, these systems are less effective against stealth attacks. Sleath attacks dispatch emails at irregular intervals specifically to evade detection.

7.2. Behavioural Heuristics

Behavioural heuristics are detection methods that analyse the characteristics and actions of an email or sender to identify behaviour that deviates from what is considered normal, legitimate, or safe. Behavioural analysis is the most effective way to detect malicious email. Some common indicators that can help are mentioned below:

7.2.1. AI-generated Content (Novel)

Attackers often use GPT-generated, slightly varied but semantically similar body texts and subject lines to trick anti-spam filters. Writing content is time-consuming. Relying on AI-generated content is more efficient for attackers, which saves time and boosts writing accuracy. Attackers often insert intended typographical or grammatical errors to lower the AI detection precision because mistakes trick AI into thinking it is human-written. Automated grammar correction tools can remove such errors efficiently before AI-content probability checks. Perplexity-based analysis, semantic similarity checks, or transformer-based detectors fine-tuned on AI-generated datasets may increase true positives of detection.

7.2.2. Irrelevant Content (Novel)

Sometimes emails are sent with randomised contents; for example, a professor who teaches arts received a research proposal about cybersecurity, and the confidential note states that she is the recipient. The conceptual area of email subject could be modelled using Latent Dirichlet Allocation (LDA) or sentence embeddings. A possible spam email may be detected in a major difference in the flow of topics.

7.2.3. Unmatched email username with name

Monitoring the sender name along with the email username is important to avoid suspicious spam messages. For example, a sender's name is John Doe, which is what is written on their account, but their email username is americano543. Such mismatched usernames often indicate spam.

7.2.4. Vague Domain Name or Extension

Examine the sender’s domain name and extension. Attackers often use unprofessional domain names and extensions to attack. For instance, "kfdsl.tech", "dadjahd.com", "testserver635.io", "top10server.com", "gmail.flowers", "outlook.jsad.com", etc. Sometimes, typographical errors are also used to impersonate professional mail servers, for example, "cafe-puskiin.ru", "gmial.com", and "icluod.io".

7.2.5. Profile Picture (Novel)

The reverse profile picture or avatar of the account analysis may reveal the originality of the sender. While monitoring, their other social media name, pictures, and activity can assist in detecting spam. Reverse image search APIs or face-matching algorithms can be used to detect stock images, archived images or cloned avatars, thereby automating the profile authenticity check. However, this detection may be time-consuming.

8. Mitigation

During a Mailbomb attack, a user cannot solely defend themselves against it. Mitigating such powerful spam attacks requires server-side and application-level advanced defence. The following controls may reduce the likelihood and impact.

8.1. Server-Side Defences

Email server administrators can adopt the following controls to prevent abuse and ensure the safety of their users:

8.1.1. Rate limiting

Restrict the number of SMTP sendings per account within a specified time range, for example, 1 email per minute for individual accounts, 10-20 emails per minute for small business accounts, depending on the policy. Thresholds should be tuned to realistic user behaviour, for instance, individuals rarely exceed 1 email/minute, while small businesses require higher limits for mailing lists.

8.1.2. Reputation checks

Validate the sender's mail server, email sending origin, domain name, and domain extension (see Section 7.2.4). This prevents impersonated domains from querying any email server. Greylist users who use anonymous services, such as VPNs or proxy nodes.

8.1.3. Malware and phishing filters

Scan attachments and embedded URLs, drop malicious emails before they are delivered to the receiver's mailbox. Attackers do not only attack with a sole purpose; they may send a malicious link or attachments (see Section 4.3).

8.1.4. Markup language changes (Novel)

Markup-structured email allows rich visual content and interactive elements. However, it can be misused and hide malicious links in target data and be masked as legitimate content, which sometimes tricks modern ML-based anti-spam filters. HTML and Markdown are the most

common markup languages used across email servers. Testing/Inventing a different markup language similar to HTML, which does not enable masking and keeps CSS styling, may reduce the chances of people being infected by hidden malicious links.

8.1.5. Volume anomaly detection (Novel)

Monitor unusual traffic for each email account and apply throttling or temporary greylisting. Temporarily restrict the receiver's incoming emails from non-whitelisted senders when they are under attack (with administrator or recipient agreement). This reduces inbox overload, ensures critical communications remain accessible, and provides time to secure or back up essential data before the attack escalates.

8.2. Application-Level Measures

Beyond server configurations, email applications can apply intelligent filtering and logic:

8.2.1. Filtering and Alerts

Set a real-time alert when the receiver's inbox is receiving abnormal email traffic. Alert them not to open any links or attachments. This prevents non-experts or decent users from falling for any scam. Filter AI-generated content directly to spam or archive folders to reduce end-user disruption and preserve server performance. Append the Geo Location metadata of the sender at the email footer to aid in detection. This feature may speed up future digital forensics and allow users to investigate.

8.2.2. Whitelisting and Verification

Auto-whitelist every email account where both parties have at least 5 sending and receiving records. It reduces the false positives of spamming from a known sender. Additionally, add a manual whitelisting feature. Adding these whitelisting filters can be faster spam monitoring while preserving the server performance. If a sender is continuously sending suspicious emails, flag their accounts and request a verification to restrict the user before sending further malicious/spam emails. The algorithm for this feature should not hinder organisations from sending newsletters or activity emails. The application may ask for business verification before allowing them to use the bulk mail feature for an extra layer of defence.

8.2.3. User and Account Policies

The email naming and username selection policy should be stricter, so that the display name matches the username, and the display name should be human-like. However, if this policy restricts users because of their names being occupied, the application can use some reserved strings before their usernames, depending on various logics.

8.2.4 Scheduled Email Operation (Novel)

Allow users to schedule a period to receive incoming emails. This may preserve performance, as well as reduce the risk of attacks when users are offline. However, this should not hinder whitelisted users from communicating when the scheduled time is over.

8.2.5. Shortened URL Handling

Shortened URLs were introduced to improve the user experience. However, some shortened URL providers do not redirect the user using HTTP 301 Redirect, making it hard to understand whether any malicious links are behind the short URL. Applications should block those shortened URLs which forbid robot access.

8.2.6. Advanced Features (Novel)

Adding a time-based or number-based batch deletion feature to clear the spam emails can be helpful to clean the inbox after any kind of spam attacks. A dedicated option for alias creation or deletion should be introduced. Let the user select their preferred alias, which may restrict the attacker from taking advantage of the common alias feature (see Section 5.2.5). For Structured-markup styled emails, consider adding a feature where the raw data/code can be shown. This assists users in understanding if any malicious links are masked.

8.3. Interception-based Email Defence (Novel)

In interception-based defence, emails are first routed through a central system before being delivered to the recipient. After passing automated filters, selected emails undergo human verification for final approval. Because suspicious emails are verified by humans after AI filtering, this hybrid approach reduces both false positives and false negatives. In Figure 5, a proposed architecture of this is visualised.

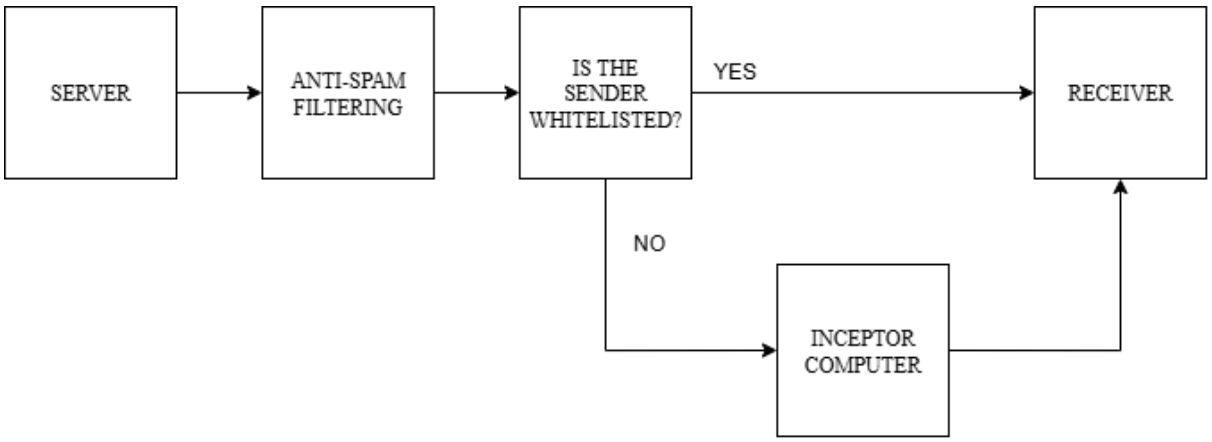


Figure 5: Proposed Architecture of an interceptor-based defence

However, this defence breaks the promise of confidentiality, as the data can be seen for monitoring purposes. Moreover, it may require one or more professional cybersecurity experts, depending on the expected email volume.

9. ETHICAL CONCERNS

All tests were conducted in isolated environments with no interaction with external users. The use of the Mailbomb script was strictly for educational and defensive research purposes. Researchers must not apply these methods to live email servers without proper legal permission.

Performing any denial-of-service operations during an armed conflict, especially if such actions disable critical infrastructure such as hospital systems and result in loss of life, may constitute a war crime under the Rome Statute of the International Criminal Court. Under Article 77, such offences may be punishable by imprisonment for up to 30 years, or life imprisonment in extreme cases.

To prevent misuse, no executable scripts have been shared in this paper. Algorithms have been presented to demonstrate the attack and develop better defence mechanisms. We disclaim any responsibility for the illegal use of the research findings.

10. LIMITATIONS AND FUTURE WORKS

This study highlights the challenges that modern ML-based anti-spam filters face against Mailbomb and other advanced spam attacks. Several limitations should be acknowledged.

First, the proposed detection and mitigation approaches were explored primarily through conceptual modelling. Real-world deployment at production-scale email servers, which handle diverse traffic volumes and user behaviours, was beyond the scope of this study. Second, financial constraints prevented the development of full prototypes for advanced strategies such as interception-based email defence and server-wide markup language redesign. Third, while this research focused on Mailbomb and spam flooding, other forms of email abuse, such as spear-phishing or business email compromise, were not extensively discussed. Finally, some policy recommendations, such as stricter username or scheduled reception of emails, may face adoption challenges due to usability or provider restrictions.

Future research could address these limitations in several ways. Interception-based email defence mechanisms would benefit from further modelling, refinement, and prototype testing. Secure delivery of styled emails with transparent link unmasking may require the development of a new markup standard that balances usability and safety.

By addressing these directions, future work can significantly enhance the defences of email systems against evolving Mailbomb and spam threats.

11. CONCLUSION

Mailbomb attacks are highly effective in resource exhaustion and inexpensive to execute. This study demonstrated that such attacks are challenging to detect by modern ML-based anti-spam filters due to their legitimate behavioural heuristics. As email continues to be reliable for future communication, understanding and preventing these attacks is essential.

ACKNOWLEDGEMENT

The author acknowledges that this research was conducted solely with self-funding, without external financial support or affiliation. The author also recognises the lasting impact of Muhammad ibn Musa al-Khwarizmi, whose work on *Al-Jabr* laid the foundations for concepts that underpin modern programming, and pays tribute to the students who lost their lives in the Milestone School Tragedy.

APPENDIX A - Email Body Generation Open AI Prompt

Please ignore all previous text. From now on, you are a writing assistant. While generating content, please ensure the following details:
1) The writing band score should not be more than IELTS 7.
2) Must use repeated words.
3) Write in passive voice sometimes. However, only use it for 20% of the content.
4) Make some grammar mistakes, such as adding 's' or 'es' for 1 sentence.
5) The logical flow should not be smooth 100%.
6) Use a limited vocabulary like a person from a non-native English-speaking country.
7) The overall text should not have 100% of the concept of the main topic.

Now, you will be given research topics, the sender's name, their capability, and the receiver's name. Each detail is sliced with " | ". Capabilities are sliced by commas. Each line contains a new detail. Write an email body within 150 words for each while following the previous instruction. The email is for a research proposal. Add "I have added my CV, transcripts, and research proposal paper in docx format." There is no need to add "Greetings, Prof." or similar things at the top. If you want, you can highlight their research, which is close to the given topic. You may also add some other achievement to improve the overall chances of reading the email.

Example Input: "Modern Algebra | Vanya Rhexis, Turkey | High school passed with GPA 5.00 (highest), got 1st merit in national talent, programming olympiad winner 2025 | Tyrea Clove, Oxford University" "Modern Air Defence | Fiie Rhexis, Turkey | High school passed with GPA 5.00 (highest), got 2nd merit in national talent, programming olympiad winner 2024 | Tyrea Clove, Oxford University"

Example Output: I hope you are doing well. My name is Vanya Rhexis from Turkey. I wish to share one of my research interests: modern algebra. This explains ..... . This research aligns closely with your ..... . and continue on writing. I hope you are doing well. My name is Fiie Rhexis from Turkey. I am interested in conducting research with you. The research topic is modern air defence. This explains ..... . This research aligns closely with your ..... . and continue on writing.

APPENDIX B - Email Username/ID to Name Generation Open AI Prompt

Please reset your memory and ignore all previous texts. You are given a large email address or username set sliced by a new line or "\n". You will be generating random and common names, regions, and their random achievements depending on the email account. While generating the name, please consider:
1) The region should match the name pattern of that region.
2) Their achievements can be volunteering- and/or olympiad- and/or online-based, depending on the topic.
3) The name should be written as FirstName LastName.

Example Input:
Topic: Computer Science, BSc
Region: Turkey [If not mentioned, then random region] [If mentioned, try to add province or state or city name.]

Email List:
v.rhexis@email.com
fie.official@email.com
Tugce Example

Example Output:
Vanya Rhexis, Istanbul Turkey | High school passed with a GPA of 5.00 (highest), got an A+ in physics and math, and was the regional programming olympiad winner in 2025.
Fiie Rhexis, Ankara Turkey | High school passed with a GPA of 5.00 (highest), got 2nd merit in national talent, and was the programming olympiad winner in 2024.
Tugce Peri, Antalya Turkey | High school passed with a GPA of 5.00 (highest) in 2024, scout leader of platoon 35, and was the programming olympiad winner in 2023 and contributed to open-source projects.

APPENDIX C - Email Subject Line Generation Open AI Prompt

Please ignore all previous text. You are a writing assistant. Take time and come up with interesting email subject lines for research proposals, which will be formally sent to a professor. Please consider:
1) You will generate the subject lines solely relevant to the topic mentioned.
2) Each email subject line should be unique and eye-catching.
3) Do not list them with bullet or numbered points.
4) Write the subject lines like official proposal submission.

You will only follow the pattern mentioned below:

Example Input:
Subject Line to generate: 2
Topic: Computer Science, Networking

Example Output:
Proposal Submission on Bluetooth Communication
Request to Collaborate: Satellite-based SOS Communication

REFERENCES

[1] “From chaos to control: Insights from 24 email bombing waves’.” https://www.xorlab.com/en/blog/from-chaos-to-control-insights-from-24-email-bombing-waves.

[2] S. Keskin and O. Sevli, “Machine Learning Based Classification for Spam Detection,” *Sakarya University Journal of Science*, vol. 28, no. 2, pp. 270–282, Apr. 2024, doi: https://doi.org/10.16984/saufenbilder.1264476.

[3] Y. Liu *et al.*, “A Systematic Review of Machine Learning Approaches for Detecting Deceptive Activities on Social Media: Methods, Challenges, and Biases,” *arXiv.org*, 2024. https://arxiv.org/abs/2410.20293

[4] N. A. Al-Dmour, Shagufta Kousar, A. Khan, Anaum Ihsan, T. Abbas, and A. Q. Saeed, “Enhancing Email Spam Detection Using Advanced AI Techniques,” *2022 International Conference on Decision Aid Sciences and Applications (DASA)*, pp. 1–6, Dec. 2024, doi: https://doi.org/10.1109/dasa63652.2024.10836525.

[5] O.T.W, “Working with exploits: Using exploit-db to find exploits - hackers arise’,” *Dec*, vol. 14, Available: https://hackers-arise.com/working-with-exploits-using-exploit-db-to-find-exploits

[6] D. Goodin, “OpenAI helps spammers plaster 80,000 sites with messages that bypassed filters’,” *Ars Technica, Apr*, vol. 09, Available: https://arstechnica.com/security/2025/04/openais-gpt-helps-spammers-send-blast-of-80000-messages-that-bypassed-filters

[7] E. Hotoğlu, S. Sen, and B. Can, “A comprehensive analysis of adversarial attacks against spam filters’,” May 2025, doi: https://doi.org/10.48550/arXiv.2505.03831..

[8] “smtplib — SMTP protocol client’, Python documentation.” https://docs.python.org/3/library/smtplib.html.

[9] “OpenAI platform’.” https://platform.openai.com.

[10] “threading — Thread-based parallelism’.” https://docs.python.org/3/library/threading.html.

[11] “proxychains-ng | kali linux tools’.” https://www.kali.org/tools/proxychains-ng

[12] “The tor project’.” https://www.torproject.org/about/history

[13] “OpenVPN Community Documentation’.” https://openvpn.net/community-docs

[14] M. A. S. B. Ahmad, M. I. B. Rozlan, and A. D. B. Yusri, “Spam Detection Using Machine Learning Techniques,” *Feb*, vol. 12, doi: https://doi.org/10.36227/techrxiv.173933255.51566942/v1..

[15] E. Blanzieri and A. Bryl, “A survey of learning-based techniques of email spam filtering,” *Artif Intell Rev*, vol. 29, Art. no. 1, Mar. 2008, doi: https://doi.org/10.1007/s10462-009-9109-6..

[16] I. B. Mustapha, S. Hasan, S. O. Olatunji, S. M. Shamsuddin, and A. Kazeem, “Effective email spam detection system using extreme gradient boosting,” doi: https://doi.org/10.48550/arXiv.2012.14430..

[17] O. Saad, A. Darwish, and R. Faraj, “A survey of machine learning techniques for Spam filtering.”

[18] J. Wang, L. Yu, J. C. S. Lui, and X. Luo, “Modern DDoS threats and countermeasures: Insights into emerging attacks and detection strategies,” *Feb*, vol. 27, doi: https://doi.org/10.48550/arXiv.2502.19996..

[19] P. Muncaster, “Russian ransomware groups deploy email bombing and teams vishing,” *Infosecurity Magazine. Accessed: Aug*, vol. 30, Available: https://www.infosecurity-magazine.com/news/ransomware-email-bombing-teams/