Abstract—This paper aims to survey various techniques utilized for content moderation in end-to-end encryption systems. We assess the challenging aspect of content moderation: maintaining a safe platform while assuring user privacy. We study the unique features of some content moderation techniques, such as message franking and perceptual hashing, and highlight their limitations. Currently implemented content moderation techniques violate the goals of end-to-end encrypted messaging to some extent. This has led researchers to develop remediations and design new security primitives to make content moderation compatible with end-to-end encryption systems. We detail these developments, analyze the proposed research efforts, assess their security guarantees, correlate them with other proposed solutions, and determine suitable improvements under specific scenarios.

1. Introduction

A big challenge in end-to-end encrypted (E2EE) messaging schemes is that it restricts the messaging scheme’s service provider from moderating content communicated through the platform. Content moderation is a vital and challenging task in today’s technologically interconnected world, where social media and the Internet have become an essential part of our communication. Content moderation is hard to achieve, especially in systems that enforce end-to-end encryption in communication. Facebook researchers introduced the idea of message franking to perform content moderation on E2EE systems (done for Facebook messenger). Message franking allows verifiable reporting of content that violates the terms and policies of the communication platform. In general, most communication platforms like WhatsApp, Facebook Messenger, etc., flag hate speech, propaganda messages, sexually inappropriate content, cyberbullying, etc., through their content moderation policies. Moreover, they heavily rely on its users’ responsible and verifiable reporting of such messages. Message franking allows for verifiable reporting of policy-violating messages sent in E2EE communication schemes while preserving deniability. Over the years, various methods have been employed to perform content moderation apart from message franking (which is known to be among the best working techniques for content moderation), such as perceptual hashing, metadata analysis, tagging, etc. The most crucial part of designing such schemes is to ensure that the users’ privacy is not compromised in the process. An adversary should not be able to obtain any additional private information after the implementation of the content moderation scheme assuming that the E2EE system has no cryptographic flaws.

1.1. Paper Outline

In this paper, we initially describe the basic terms associated with Content Moderation, the process and stages of content moderation, and the goals related to it. Then, we study the related work and analyze the contributions made by other authors in this area. Section 3 discusses various schemes that are currently available, and we explore the pros and cons associated with each scheme. We summarize how every scheme works and the goals that these schemes try to achieve. Finally, we compare all the content moderation schemes that we surveyed previously and map their goals to the three goals of content moderation schemes - privacy, accountability, and deniability, and comment on their detection efficiency.

1.2. End-to-End Encryption

Traditionally, encryption means a scheme that takes a key and a message as input to output a ciphertext. The plaintext is recoverable if and only if the key is known. The algorithm for both encryption and decryption is typically in the open domain, with only the key being the unknown element to an adversary. For two users communicating over a messaging platform, the encryption keys are typically stored on the server that’s managed by the platform hosting entity. Therefore, the hosting entity can decrypt the messages sent by its users to one another. End-to-end encryption takes this to a higher level where only the communicating users can read the message. Most messaging platforms have evolved over the years to make their platform privacy-preserving. Therefore, they have switched to E2EE. As a result it prevents potential eavesdroppers - telecom providers, ISPs (Internet Service Providers), malicious adversaries, and even the hosting provider itself - from being able to read messages and access the cryptographic keys needed to decrypt the messages.
communication. The sender encrypts the messages, and the recipients retrieve the data and decrypt it themselves.

A big challenge that systems face when incorporating E2EE in their systems is to hand over the communication of its users to the governing authorities when requested due to the infeasibility of reading the communication between the sending and the receiving parties. This also results in a challenge to enforce content moderation and have E2EE. Although E2EE is not perfect (several potential attacks have been proposed), it serves to provide a decent amount of security and privacy. E2EE is used in messaging platforms, storage systems, and email services. An E2EE-supported storage system stores files/photos on cloud servers in E2EE format. The data is encrypted using a key known only to the designated user. The user is the only party guaranteed to have access to the data. Keybase uses this approach to securely persist user files in the cloud. Messaging platforms like Signal and WhatsApp encrypt the messages with keys known only to the two users. WhatsApp and Signal typically generate a fingerprint produced from the keys that allow users to verify their keys with each other. This form of authentication guarantees safety against MiTM attacks in the system. Emails can also be end-to-end encrypted where even the email server cannot read the contents of the email.

1.3. Content Moderation

As per [1], content moderation is the set of policies and systems that decide what kind of user-generated content by accounts is to be retained, deleted, or modified on a communication platform that the account holders of the platform use. Every system that hosts a platform typically has its set of guidelines or rules that every account holder must follow. These rules can be either directly set by the system, and they may be in accordance with the governing body in which the platform is running. Based on the content the account holder is posting on the platform, the platform’s system decides to moderate the posted content and checks whether the content complies with their guidelines.

There are a variety of approaches for performing content moderation. Typically, this is done with automated systems capable of detecting and analyzing videos, text, and photography and checking for potential copyright infringement, pornographic content, etc., before it has been posted on the platform. Various Machine Learning algorithms majorly aid in achieving this. There exist semi-automated content moderation techniques as well where the automated system handpicks certain content that it “thinks” would violate the guidelines and sends it to further review to a human moderator. There also exists a manual system that has trained professionals hired to moderate the content. More minor services that don’t have a large user base prefer to do the moderation manually, while extensive services like Facebook, YouTube, etc., have a semi-automated content moderation approach.

Additionally, these content moderation decisions change from region to region. Apart from the standard guidelines that the system has set, they must also comply with the governance guidelines for that particular region. These governance guidelines can dynamically change if a specific event takes place that the government wants to be moderated. For example, Twitter was blocking tweets regarding the January 6th, 2021, attacks at the US Capitol for 24 hours [2].

Some systems set their guidelines and rely on the user communities to do the content moderation on behalf of the system. For example, Reddit has a concept called “subreddits” and each “subreddit” has its own set of guidelines that the users of the “subreddit” are required to follow. These are an additional set of guidelines that users must follow apart from the standard ones that Reddit sets [3].

1.4. Process of Content Moderation

The process of content moderation starts with defining the guidelines of the system. The hosts decide which user-produced content is not permitted on the platform. These guidelines are publicly available, and users must agree to these guidelines before signing up for the service. A typical checkbox that appears on every sign-up page of a social media platform has a link to an elaborate legal document that states these guidelines along with other legal terms associated with the system [4].

The next step is to detect such user-generated content that violates the system’s guidelines. Many of these detection schemes are privacy-invasive since it requires access to the actual data content. Now, for social media platforms, the process is relatively straightforward since most of the content is available publicly / to a subset of “peers”. However, this becomes challenging once E2EE environments come into consideration. This includes systems like WhatsApp, Instagram DMs, iMessage, etc. Several metadata parameters consider when such E2EE systems are to be moderated for violating content. IP address, account-related information, frequency of posting, etc., prove useful parameters when the content moderation schemes for such E2EE environments are used. Typically this is seen more often in coordinated schemes that involve several user accounts that are either compromised, or bot-accounts [5].

After this step, the user-generated content is checked for violations. It is also checked to determine whether it violates the policies or any federal law. Typically, when automated content filtering is employed to block content at upload, the steps of detection and evaluation can be merged into one.

Once the content is checked for violations, “enforcement” is performed against the violated user-generated content. There are several ways to handle this, with the easiest and the most popular one being - removal of the content. For example, tweets sent during the Presidential Election of 2020 had a warning if they were fact-checked to be untrue by the moderations team. Twitter uses a scheme of “shadowbanning” users if they post such kind of content that reduces the visibility of such content. Accounts also get suspended / removed for violation [6].

Not all, but some systems allow users to appeal for a review of the content moderation decisions that the users
think are incorrect. Since the systems are automated, they are prone to errors. Appeals assist in fixing such errors. Finally, the last phase is about educating users about the content moderation policies and how they can be enforced. These are typically seen through the terms-of-service check-boxes, community guidelines, and moderators’ guidelines. Most of the platforms often tend to provide reasonings for violations referencing their terms which also comes under the “educating” phase.

### 1.5. Goals of Content Moderation

With the design of a content moderation scheme, we have the following goals -

1) Privacy: The messages reported by the recipient of abusive messages should be the only ones that are disclosed, and any other messages involving the recipient should be private by default.

2) Accountability: The end-to-end encrypted system must be able to verify the sender of the abusive messages when the recipient reports them.

3) Deniability: The report of abusive messages raised by the recipient must only be verifiable by the moderator of the end-to-end encrypted system.

These goals apply to any content moderation scheme regardless of whether it is used for E2EE.

### 2. Related Work

Various studies on content moderation techniques have been conducted over the past decade.

Mayer et al. showed that content moderation could be carried out without having to comprise user privacy in end-to-end encryption systems [7]. Law enforcement intervention may not be necessary to recover the contents of messages and deem them malicious. Secure computations based on cryptographic primitives can lead to constructions of schemes that satisfy the main goals of content moderation: privacy, accountability, and deniability.

Kamara et al. studied the existing technical proposals designed for content moderation in end-to-end encryption systems to find out the most viable approaches that efficiently conduct content moderation without having to violate the security, and privacy goals of the platform [4]. After describing various proposals for content moderation, the authors found that user-reporting and meta-data analysis are the two most effective tools for detecting inappropriate content while ensuring that the privacy and security guarantees of the end-to-end encryption system are met. The authors also found that traceability and perceptual hashing methods for content moderation fail to practically meet the goals of end-to-end encryption systems. They state that machine learning classifiers could be an effective tool for content moderation, but it requires more research.

These studies, however, do not present concrete cryptographic notions of security for analyzing the schemes and proving their claims. Moreover, a comparative study of content moderation conducted on different communication platforms that use the presented approaches would lead to understanding the implementation-specific factors that could potentially contribute to the effectiveness of the methods. We attempt to extend their work by understanding the underlying cryptographic primitives in content moderation techniques. We survey the available literature on these techniques to develop a comprehensive understanding of the notions of security provided by the methods, possible threats and attacks on the schemes, present implementations, and future directions concerning the techniques.

### 3. Content Moderation Schemes

#### 3.1. Message Franking

Message franking is an abuse reporting mechanism where a user can report malicious behavior on end-to-end encrypted communication platforms. It is designed to protect the anonymity of the reporter while supporting the verification of their identity. Moreover, framing a user by crafting fake reports is not possible. The main challenge in moderation of end-to-end encrypted systems using message franking is guaranteeing accountability while preserving deniability. The principle of deniability was a fundamental goal of Facebook’s message franking [8].

Message franking was first studied by Grubbs et al.,. They proposed a new cryptographic primitive, compactly committing authenticated encryption with associated data for capturing the notion of security in message franking [9].

A small part of the ciphertext is used as a cryptographic commitment of a message in this authenticated encryption with associated data (AEAD) scheme. The responsibility can be opened by decrypting the message. They prove that efficient message franking schemes can be constructed by the addition of secret keys that would function as an opening mechanism for the commitments in existing AEAD schemes. Their result demonstrates that many existing symmetric encryption schemes can be called committing in the traditional sense.

Facebook’s message franking protocol. Facebook’s protocol works as follows -

1) An HMAC key is generated by the sender. The HMAC for the message is generated.
2) An AEAD scheme is used to encrypt the message along with the HMAC key, and the ciphertext and its hash are sent to Facebook’s servers. The key used is a symmetric key shared between the sender and the recipient.
3) Facebook then signs the received hash and sends the produced signature, HMAC hash, and the ciphertext to the recipient.
4) The recipient decrypts the ciphertext and verifies the HMAC using the HMAC key.
5) If the recipient believes the message to be malicious, it can be reported by sending the signature,
message, HMAC hash, and HMAC key to Facebook.

6) Facebook verifies the received signature and hash for the reported abuse.

Here, HMAC plays the role of a commitment. This is unlike a committing encryption scheme where the entire ciphertext is a commitment, which, when decrypted, reveals the key. In the message franking protocol developed by Facebook, only a part of the ciphertext acts as a commitment to the message.

Dodis et al. demonstrated an attack on Facebook’s attachment franking scheme, which uses AES-GCM for the franking of multimedia attachments [10]. The attack makes reporting of abuse of an attachment to fail. An adversary can send a malicious attachment in the form of an image or video to an unsuspecting user, and the user would not be able to successfully report the abuse. A benign attachment is sent to Facebook instead of the actual malicious one. A cryptographic hash of the AES-GCM ciphertext and a randomly generated value are used to identify an attachment uniquely. The attack relies on finding two keys and a ciphertext, where the first key can decrypt the ciphertext to a malicious attachment and the second key can decrypt the ciphertext to a benign attachment. The adversary sends two messages with these two keys to the recipient along with the ciphertext attachment. When the recipient reports abuse, Facebook’s servers deduplicate the two attachments, and the benign attachment is the only one that is reported. The authors developed the fastest compactly committing AE scheme and introduced a new primitive called encryption to capture binding security guarantees that AES-GCM did not provide. Encryption ensures that the one-time real-or-random notion of security and second-ciphertext unforgeability (SCU), a variant of one-time ciphertext integrity that requires forging a ciphertext for a given binding tag with a known key, are satisfied. The authors also demonstrate how encryption can be used to build ccAEAD schemes.

The improvements mentioned above to Facebook’s message franking protocol and its building blocks did not account for one major pitfall. Reporting a single abusive message result in disclosing the entire communication for a session. This violates user privacy. The adversary can aggrivate the situation by intentionally sending malicious messages in a conversation that includes the victim’s private information. Reporting the abuse would result in revealing the private information. Hence, the user may not choose to convey the malicious message to protect their privacy. Chen et al. address this pitfall by introducing and formalizing a new primitive called targeted opening compactly committing AEAD (TOCE) [11]. The abuse reporting using this reporting allows the victim to selectively disclose a subset of the conversation and keep the rest of the conversation discrete. The desired notions of target hiding, position binding, and efficient checkable for a targeted opening commitment scheme (TOC) are defined.

Leontiadis et al. contributed towards solving this problem by providing a similar security definition called After Opening Privacy (AOP) that preserves the privacy of the non-abusive contents in a reported message [12]. They provide two constructions that require at least four pass block cipher operations. The constructions provided by Chen et al. require at most three passes block cipher operations and consider other properties, such as integrity, sender binding, and receiver binding under the targeted open capability. Leontiadis et al. also introduce a new definition for multi-opening indistinguishability with a partial opening (MO-IND-PO), which forces an adversary to distinguish encryptions of abusive blocks. This revision to the security definition for multi-opening ciphertext integrity was introduced by Grubs et al. .

The message franking protocol would become ineffective when implemented on platforms like Signal, where user identities are hidden because the protocol requires user identifiers to be extracted from encrypted messages. Tyagi et al. proposed an asymmetric message franking scheme (AMF) to solve this problem where the user identities and metadata remains hidden from the end-to-end encrypted platform. AMF uses three algorithms for accountability: Frank, Verify and Judge, and three for deniability: Forge, RForge, JForge. The adversary would sign the message using Frank and provide the ciphertext of the message to the recipient. The recipient would recover the plaintext and verify the received signature using Verify. If the recipient wishes to report the message, the message and signature would have to be sent to the moderator. The moderator would use the judge to verify the report. The deniability algorithms are defined for capturing the forging capabilities of the adversary. The authors provide security notions for accountability: unforgeability, sender binding, and receiver binding. They introduce various notions of deniability. Universal deniability requires that any non-participating party (no access to sender, receiver, or judge’s secret keys) can forge a signature indistinguishable from honestly-generated signatures to other non-participating parties that would allow the sender to claim a message originated from any non-participating party. Receiver compromise deniability requires that a party with access to the receiver’s secret key can produce the forgery of a signature that is indistinguishable from honestly-generated signatures to other parties having access to the receiver’s secret key. Judge compromise deniability requires that a party with access to the judge’s secret key can produce the forgery of a signature that is indistinguishable from honestly-generated signatures to other parties with access to the judge’s secret key. The Forge, RForge, JForge algorithms are used for universal deniability, receiver compromise deniability, and judge compromise deniability. The authors construct and study an AMF scheme and provide benchmarking results by implementing the scheme to test its performance.

3.2. Message Traceability

A common concern in E2EE systems is identifying which users have shared content that has violated the terms
of the system. A proposal by Tyagi, Miers, et al. is based on the idea of message franking, as seen previously. It allows the system to track all the users who forwarded or received the content. Before even anything is flagged, the message is kept confidential, and only the sender and recipient of the message can see the contents of the message. However, once it gets reported, it allows the system to find the messages with the same content that weren’t reported before. This allows the system owners to expose the privacy of the senders and recipients in the forward chain. If user A sends a message to user B and then user B forwards the same message to user C, and if user C reports this message, then the system host can not only know that the message was sent from B to C but also know that A sent initially it to B before it was sent to C. Just by one user reporting the message, all the previous recipients and the sender can be exposed from the chain.

This traceability feature is typically helpful in preventing misinformation from spreading on social media platforms. Even though there are ways for users to “fact-check” information, finding the source and the recipients of such messages is also crucial. WhatsApp recently added checks like preventing a multiple-times forwarded message from being shared with more than five people and adding a search button next to a multiple-times forwarded message with factual information that searches the text of the message on the Internet for factual accuracy. While these features suppress the spread of misinformation, finding the origin is handled through traceability techniques.

One way to handle this is to add the sender’s encrypted identity information to the metadata of every message. As soon as the message is reported, the recipient and sender can be retrieved from the metadata. Another way can be for providers to maintain a hash database of all the messages sent on the platform. The hash can be matched against the database to find the sender whenever inappropriate content is found.

Although these techniques might sound helpful, they are practically infeasible and have several privacy flaws. It reveals the originator’s identity to a third-party service that essentially violates the principles of E2EE. Traceability is not consistent with the principles of E2EE. Significant research needs to take place in this area to allow the correct levels of traceability without violating the tenets of E2EE. Currently, none of the E2EE systems employ message traceability.

While confidentiality and traceability may seem like they are two conflicting terms, some forms of tracing can be implemented in a non-coercive way that does not violate the confidentiality principle. For instance, consider the message flow from A to B, B to C, and finally C to D. If D reports the message and B and C agree to cooperate with D. The message flow can be traced back to A. Nobody is breaking E2EE principles since each participant is an authorized recipient of the message, and every participant knows where they received the message from. This form of cooperative tracing does not violate the E2EE ideology.

Two critical questions to answer here are - Precisely, how much information will recipients be required to reveal even though they do not wish to and who will be able to access this information? We studied two proposals that add traceability to E2EE communication schemes, and both of them have similar assumptions. These proposals require making modifications to the client software that is used for communication to ensure that the content tracing information is stored as “escrow” at the provider every time content is sent from one user to another.

In one of the proposals, as seen in [13], every time someone sends a message to another user, a new encryption key $K$ is generated. The sender uses this key to encrypt a message record with the content / the hash of the content and the sender and receiver identities. This is stored on the message server as a kind of “key escrow”. The key $K$ is not sent to the server. The sender transmits the record encryption key $k$ to the recipient using end-to-end encryption. When the next user forwards the same content to a different user, the same process is repeated, as explained before, and at that point, all the keys generated by its previous users will also be sent. Every time the entity that wants to moderate the content will have a list of encryption keys that correspond to everyone in the chain of messages for that content. They can obtain the encrypted records from the servers and use the chain of keys to decrypt them, which will reveal the forwarding path, thereby allowing us to backtrack to the origin. The flow of messages from A to B and then to C is seen in Fig. 1.

The other proposal requires every person who originates new content into the network to “watermark” the content that identifies the account ID of the sender. The watermark can be encrypted using a key stored on the servers. Therefore, this tracing scheme requires the provider’s involvement.

### 3.3. Metadata Analysis

Metadata analysis is typically used extensively in spam filtering. The metadata about a file, such as the type of
file, its size, creation time, access time, last modified field, sender and receiver information, can reveal a significant amount of information about the file, which can be enough to distinguish whether the file contains malicious content or not. ML models can be trained to derive attributes from metadata to classify the contents present in the files. They can also automatically observe user behavior on the platform to detect malicious and anomalous activities such as sending spam messages. Differentiating between a real human user and a controlled bot is another use case for ML models.

Greschbach et al. studied the privacy implication when metadata analysis is used as a content moderation technique on decentralized end-to-end encrypted communication systems [14]. In centralized systems, the metadata processing would be carried out on a single node, a single point for failure, and data aggregation. Trust is placed on this node, and caution needs to be taken to ensure that privacy and security are not violated when operations are being conducted on this node. On the other hand, decentralized systems have an infrastructure where data handling is undertaken on various nodes. Decentralized systems require the need for placing trust in all the participating nodes. Analyzing the intercommunication between the nodes leads to a potential for violating the privacy of the communicated metadata. Overall, the attack space is widened. Approaches such as padding for hiding the ciphertext size, obfuscating metadata fields, using attribute-based encryption, and obfuscating information flow for protection from traffic analysis attacks can be employed to protect the users’ privacy by making metadata retrieval and extraction of relevant information from metadata a problematic task.

3.4. Perceptual Hashing

Matching models is a content moderation technique that can detect malicious content by analyzing existing and known images and videos and matching them with the suspected images and videos. It performs well when malicious content having similar characteristics with minimal variations is spread on the end-to-end encrypted platform. Matching models include cryptographic hashing, and perceptual hashing [15]. Perceptual hash functions such as Microsoft’s PhotoDNA or Facebook’s PDQ generate a digest of the suspected image, which is then matched with a database of existing and known digests obtained from malicious images. The digests generated by perceptual hashing algorithms are identical when the underlying images have similar characteristics, undergo resizing or re-encoding, or overall appear similar to a human eye. Therefore, they effectively capture unknown malicious content that is similar but not precisely identical to known malicious content. Cryptographic hashing techniques can detect identical content. The scanning in the end-to-end encrypted system for malicious content can be performed on the server-side or the client-side. Server-side scanning involves hashing the content, encrypting it, and sending the hashes to the servers for matching. This approach discloses the content hash, which harms privacy as some information may still be disclosed by the hashes, specifically when the hash is used to check whether two images are identical. Client-side scanning involves storing the database of malicious content hashes on the end-devices of the user of the end-to-end encrypted system.

Prokos et al. built threat models around perceptual hashing algorithms considering an adversary’s perspective, such as framing and censorship, detection avoidance, user data leakage, and illicit-content data leaks [15]. They presented attacks against PhotoDNA and PDQ. Their results suggest that targeted second-preimage attacks, where an adversary produces a variant of an image collides with a targeted digest but goes undetected, can be efficiently deployed. This indicates the insufficiencies in existing perceptual hash functions that can harm the security and privacy of users in end-to-end encryption systems.

Jain et al. demonstrated the vulnerability of perceptual hashing-based client-side scanning mechanisms to detect avoidance attacks by deploying their attack on pHash (continuous and discrete), dHash, aHash, and PDQ algorithms [16]. Detection avoidance attacks allow an adversary to minimally modify an existing malicious content (which is present in the database of malicious contents) such that its content is preserved. Still, it goes undetected and is not flagged by the end-to-end encryption system. The attack should still be possible even when the platform takes defensive measures such as expanding the database to include other minimally modified images.

Perceptual hashing was proposed as a mechanism for detecting misinformation on Whatsapp by Reis et al. Messages sent on Whatsapp are fact-checked, and messages containing misinformation are recorded in a database through perceptual hashing [17], and the database is stored on the client-side. The spreading of disinformation can be detected at the receiver by matching the message hashes in the database. The receiver is informed about the misinformation by flagging the message.

3.5. NeuralHash - Perceptual Hashing Technology by Apple

On August 5th, 2021, Apple announced a technological measure under the umbrella term called “Expanded Protections for Children” to safeguard its users from child sexual abuse material. The proposed technology works by continuously monitoring photos saved or shared on Apple devices. The authorities are alerted if any CSAM-related objectionable images are detected in iCloud (Apple’s cloud storage system). The child’s parents are also notified depending on whether iMessage was used to send/receive photos their AI algorithm thinks are inappropriate. As per Apple, these checks are performed on-device [18]. As per Apple’s technical documentation, the technology is capable of providing the following security and privacy measures -

1) Apple does not know images that do not match the CSAM database.

2) Metadata or visual derivatives for matched CSAM images cannot be accessed unless a threshold of matches is exceeded for the account.
3) Users cannot access or view the database of known CSAM images.
4) Users cannot identify which images are flagged as CSAM by the system.

This technology uses PSI (private set intersection) along with their perceptual hashing mechanism, which is termed as “NeuralHash”. NeuralHash can analyze an image and convert it to a hash value specific to that image. An image that appears to be nearly identical will produce the same number. The algorithm can consider pictures that differ in size or have filters/transcoding applied to them, and it would still manage to give the same hash value. As seen in Fig. 2, before an image gets uploaded on their photo storage system called iCloud Photos, the NeuralHash algorithm runs locally. It produces a hash value mapped to a known CSAM hashes database. This is done using Apple’s implementation of a private set intersection. PSI allows for matching without revealing the result of the intersection outcome. Once PSI is implemented, a secure “voucher” is created that encodes the matched results and encrypts the NeuralHash value, and a visual derivative of the image [19]. This “voucher” is then securely uploaded to iCloud Photos as a part of the image’s metadata along with the image. Through the cryptographic primitive of threshold secret sharing, the contents of the safety vouchers are protected from Apple unless the account crosses a threshold of CSAM content.

![Diagram of CSAM Detection Scheme in Apple Devices](image)

Figure 2. CSAM Detection Scheme in Apple Devices

This example is an E2EE-based cloud-storage solution that uses content moderation schemes. Several researchers have found flaws and issues in the proposed “NeuralHash” algorithm. Security researchers could find hash collisions for images that were nowhere close to each other. Several adversarial examples were demonstrated that managed to hoodwink the NeuralHash algorithm. Researchers also stated that this technology would eventually, turn every Apple device into a continuously scanning system that can be a radically new tool for invasive surveillance through flaws in the technology.

3.6. Effective Private Membership Computation

This is a content moderation technique that allows perceptual hashing without revealing the underlying message or hash to the server. The server only knows whether the content matches a record in the known and existing malicious content database. Additionally, the content of the database remains unknown to the user. Kulshreshtha et al. provide constructions for such privacy-preserving perceptual hash matching, private exact membership computation (PEMC) and private approximate membership computation (PAMC) [20].

3.7. Predictive Models

Prediction models are machine learning-based techniques used to differentiate between malicious and benign content for content moderation in end-to-end encryption systems. It involves training the ML models that include computer vision models for finding the shapes, colors, and textures in images and computer audition models for finding characteristics and attributes in audio files. Convolutional neural networks (CNNs) are typically used in this process. Currently, efficient techniques that use predictive models are not developed. Feeding labeled data to train the models is an intensive task. Running the models on the client-side would preserve the privacy of the multimedia file being tested for malicious content. Still, the performance would depend on the hardware and software capabilities of the user’s device. Server-side verification may be better in performance, but it would raise privacy concerns. Privacy-preserving predictive models could be developed to address this issue.

Shenkman et al. enlisted various algorithms used in content moderation using predictive models. They provide insights about the type of content present in images and are described as follows:

1) Classifiers: These are the most common algorithms used in content moderation. They can determine the contents of an image.
2) Object detectors: It extends the capabilities of a classifier to localize and classify different objects present in an image.
3) Semantic Segmentation and Instance Segmentation: This allows for establishing a relationship between the objects present in an image. Semantic segmentation involves labeling every pixel in an image. Instance segmentation involves identifying individual object boundaries.
4) Scene understanding: An understanding of the scene is built by analyzing the geometric and semantic relationships and correlations between the images’ objects.
5) Object tracking: The course of an object in a video is traced to understand the content in a video.

The predictive models usually give better results on more straightforward objective tasks such as determining whether an object is present in a given image [21]. They work better as building blocks, and their reliability decreases as the task of interpreting the images become complicated. They lack in accurately determining the context of a multimedia file. Existing techniques are not robust in processing variations of an image which are produced to avoid detection [7].
4. Comparison of the Content Moderation Techniques

In Table 1, we provide the comparison between the presented content moderation techniques on the basis of privacy, accountability, deniability, and detection. We note the contributions done on the original techniques and build our assessment after considering the suggested improvements.

5. Conclusion

This paper studied several content moderation schemes used on messages, cloud storage, and email communication in end-to-end encrypted systems. We summarized and analyzed the pros and cons and design details of these open-design implementations of the solutions now implemented by these E2EE systems in their products. Several of these solutions had other privacy implications, and some partially violated the principles of E2EE design. However, all of them are researched in different directions to achieve content moderation in such schemes by adhering to privacy, accountability, and deniability, as stated previously. Apple’s NeuralHash perceptual hashing scheme suffered from several flaws, and the privacy guarantees they had initially assumed weren’t satisfactory. Even though many of these ideas are now implemented in production, studying their security and privacy guarantees from a formal standpoint and looking for practical loopholes is vital. Looking at it from all perspectives is essential for any scheme proposed in the future, as seen in Apple’s case study. Overall, we did a comprehensive survey of the available content moderation schemes through various research papers and attempted to provide a systematic and thorough look at the current research and future directions in this area.

References

[1] Hannah Bloch-Wehba. Automation in moderation. *Cornell Int’l LJ*, 53:41, 2020.
[2] Reuters. Twitter on alert for calls to violence on us capitol attack anniversary, Jan 2022.
[3] Reddit’s content policy, 2022.
[4] Seny Kamara, Mallory Knodel, Emma Llans, Greg Nojeim, Lucy Qin, Dhanaraj Thakur, and Caitlin Vogus. Outside looking in: Approaches to content moderation in end-to-end encrypted systems. *arXiv preprint arXiv:2202.04617*, 2022.
[5] Camille Francois. Actors, behaviors, content: A disinformation abc. *Algorithms*, 2020.
[6] Francisco S Marcondes, Adelino Gala, Dalila Duraes, Fernando Moreira, Jose Joao Almeida, Vania Baldi, and Paulo Novais. A profile on twitter shadowban: An ai ethics position paper on free-speech. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 397–405. Springer, 2021.
[7] Jonathan Mayer. Content moderation for end-to-end encrypted messaging. *Princeton University*, 2019.
[8] Stefania Andries, Andrei-Daniel Miron, Andrei Cristian, and Emil Simion. A survey on the security protocols employed by mobile messaging applications. *Cryptography ePrint Archive*, 2022.

| Technique                  | Privacy                  | Accountability               | Deniability                        | Detection                           |
|----------------------------|--------------------------|------------------------------|------------------------------------|-------------------------------------|
| Message Franking           | The notions of TOCE and AOP guarantee privacy. | The notions of encryption and compactly committing AEAD guarantee accountability. | Asymmetric message franking schemes guarantee deniability. | Relies on user reporting |
| Message Traceability       | Strong accountability is provided. | Strong accountability is provided. | Weak deniability is provided. | Detection works without any issues if the keys-based mechanism as explained in the section on Message Traceability is applied correctly. |
| Metadata Analysis          | Privacy is not guaranteed. Potential to violate privacy is high in case of decentralized processing of metadata in comparison with a centralised architecture. | Strong accountability is provided. | Weak deniability is provided. | Inferences derived from metadata may not always be possible. Reliability would depend on the qualities of the trained ML models. |
| Perceptual Hashing         | Privacy is provided when privacy-preserving perceptual hashing techniques such as private membership computation are used. Client-side processing is preferred over server-side. | Strong accountability is not provided due to the feasibility of targeted-second-preimage attacks. | Strong deniability is provided. | Accurate detection is not provided as detection avoidance attacks are possible. |
| Predictive Models          | Strong notions of privacy are not provided. Client-side processing is preferred over server-side. | Strong accountability is assumed and not proven. Attacks may exist. | Strong deniability is assumed and not proven. Attacks may exist. | Detection of malicious content is not reliable nor performant. |

**TABLE 1. COMPARISON OF CONTENT MODERATION TECHNIQUES ON THE BASIS OF PRIVACY, ACCOUNTABILITY, DENIABILITY AND DETECTION**
[9] Paul Grubbs, Jiahui Lu, and Thomas Ristenpart. Message franking via committing authenticated encryption. In Annual International Cryptology Conference, pages 66–97. Springer, 2017.

[10] Yevgeniy Dodis, Paul Grubbs, Thomas Ristenpart, and Joanne Woodage. Fast message franking: From invisible salamanders to encryptment. In Annual International Cryptology Conference, pages 155–186. Springer, 2018.

[11] Long Chen and Qiang Tang. People who live in glass houses should not throw stones: targeted opening message franking schemes. Cryptology ePrint Archive, 2018.

[12] Iraklis Leontiadis and Serge Vaudenay. Private message franking with after opening privacy. Cryptology ePrint Archive, 2018.

[13] Nirvan Tyagi, Ian Miers, and Thomas Ristenpart. Traceback for end-to-end encrypted messaging. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, pages 413–430, 2019.

[14] Benjamin Reschbach, Gunnar Kreitz, and Sonja Buchegger. The devil is in the metadata—new privacy challenges in decentralised online social networks. In 2012 IEEE international conference on pervasive computing and communications workshops, pages 333–339. IEEE, 2012.

[15] Jonathan Prokos, Tushar M Jois, Neil Fendley, Roei Schuster, Matthew Green, Eran Tromer, and Yinzhi Cao. Squint hard enough: Evaluating perceptual hashing with machine learning. Cryptology ePrint Archive, 2021.

[16] Priyanka Samanta and Shweta Jain. Analysis of perceptual hashing algorithms in image manipulation detection. Procedia Computer Science, 185:203–212, 2021.

[17] Julio Reis, Palhe de Freitas Melo, Kiran Garimella, and Fabricio Benevenuto. Can whatsapp benefit from debunked fact-checked stories to reduce misinformation? arXiv preprint arXiv:2006.02471, 2020.

[18] Apple Inc. Csam detection technical summary, Aug 2021.

[19] Abhishek Bhownick, Dan Boneh, Steve Myers, Kunal Talwar, and Karl Tarbe. The apple psi system, 2021.

[20] Anunay Kulshrestha and Jonathan Mayer. Identifying harmful media in {End-to-End} encrypted communication: Efficient private membership computation. In 30th USENIX Security Symposium (USENIX Security 21), pages 893–910, 2021.

[21] Carey Shenkman, Dhanaraj Thakur, and Emma Llanós. Do you see what i see? capabilities and limits of automated multimedia content analysis. arXiv preprint arXiv:2201.11105, 2021.