USENIX

THE ADVANCED COMPUTING
SYSTEMS ASSOCIATION

ATKSCOPES: Multiresolution Adversarial Perturbation as a Unified Attack on Perceptual Hashing and Beyond

Yushu Zhang, Yuanyuan Sun, and Shuren Qi, *Nanjing University of Aeronautics and Astronautics;* Zhongyun Hua, *Harbin Institute of Technology, Shenzhen;* Wenying Wen and Yuming Fang, *Jiangxi University of Finance and Economics*

https://www.usenix.org/conference/usenixsecurity25/presentation/zhang-yushu

This paper is included in the Proceedings of the 34th USENIX Security Symposium.

August 13–15, 2025 • Seattle, WA, USA

978-1-939133-52-6

Open access to the Proceedings of the 34th USENIX Security Symposium is sponsored by USENIX.

ATKSCOPES: Multiresolution Adversarial Perturbation as a Unified Attack on Perceptual Hashing and Beyond

Yushu Zhang[1], Yuanyuan Sun[1], Shuren Qi[1,*], Zhongyun Hua[2], Wenying Wen[3], Yuming Fang[3]

[1]Nanjing University of Aeronautics and Astronautics
[2]Harbin Institute of Technology, Shenzhen
[3]Jiangxi University of Finance and Economics
[*]Corresponding author: Shuren Qi, Email: shurenqi@nuaa.edu.cn

Abstract

Privacy and regulation are a long-lasting conflict in modern instant messaging, where the security community attempts to bridge this gap from a technological perspective. End-to-end encryption (E2EE) is a mathematically guaranteed *privacy policy* that has been widely built into commercial instant messaging applications. On the other hand, regulatory designs compatible with E2EE privacy are severely restricted, i.e., content auditing is (almost) impossible on ciphertext. For this reason, the community develops perceptual hash matching (PHM) as a *regulation policy*, where content-aware hash codes for media are computed prior to E2EE and matched against known criminal media, e.g., child pornography images, on the server side.

In this paper, we systematically reveal a range of adversarial threats to such E2EE-PHM systems, leading to regulatory failures. Unlike previous case studies, our attack is a more realistic threat – *uniformly fooling* the famous `pHash`, Facebook `PDQ`, Microsoft `PhotoDNA`, and Apple `NeuralHash`, even with *higher success rates* and *less training rounds*. Here, we validate the above proposition in both scenarios of *escaping* and *triggering* regulation.

Our main contribution is a new idea of *multiresolution perturbation*, where each perturbation element can affect image regions of adjustable scales. With this new idea and its well-formalized design, our attack encapsulates previous attacks as special cases – in some scenarios, it exhibits a huge leap in convergence efficiency compared to previous ones. Based on the above technical insights, we also discuss possible countermeasures and recommendations for social good.

1 Introduction

1.1 Privacy v.s. Regulation

With the popularity of modern instant messaging, concerns about the potential compromise of personal privacy have become increasingly prominent. In response to *privacy concerns*, the *end-to-end encryption* (E2EE) has emerged as a crucial safeguard in instant messaging applications [42].

Currently, more than 2 billion people around the world use messaging platforms that support E2EE [50], such as WhatsApp[1], Telegram[2], and Messenger[3]. Here, E2EE ensures messages, photos, videos, and beyond are protected from unauthorized access – encrypting the data so that only the intended recipients can access the content, preventing even the platform from accessing it [42].

However, governments and agencies [34, 35] have expressed *regulation concerns* – E2EE will prevent the regulation of content that violates basic human rights, such as child pornography, terrorism, drugs, and violent crime. Here, the platform is unable to access plaintext or perform decryption, which limits the regulation ability to review and account for criminal behavior.

In response to regulation concerns of E2EE privacy, many platforms [2, 3, 33] have recently proposed deploying *perceptual hash matching* (PHM) to detect criminal content. With such PHM regulation, each visual media file uploaded by the user is first processed using a perceptual hash function to generate a fixed-length data fingerprint (hash code). The resulting hash code is then matched against all known criminal media files in the database of the platform.

Unlike traditional cryptographic hash of collision resistance, PHM systems are designed to perform *non-exact* matching – visually similar media files can be matched with also similar hash codes, measured by distance functions. Once a match occurs, the potentially criminal user is automatically recorded, where the potentially criminal media may be reported to an auditor (machine or real person) for further verification.

[1]https://faq.whatsapp.com/820124435853543.
[2]https://core.telegram.org/api/end-to-end.
[3]https://about.fb.com/news/2023/12/default-end-to-end-encryption-on-messenger/

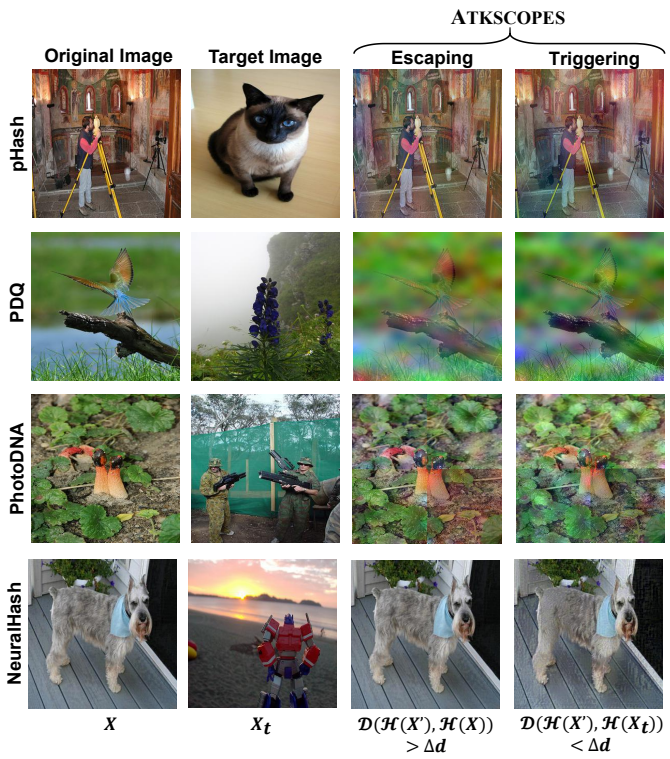

Figure 1: Illustration of the proposed ATKSCOPES for famous pHash, Facebook PDQ, Microsoft PhotoDNA, and Apple NeuralHash algorithms. With our ATKSCOPES, the listed attack images can all escape or trigger regulation, while maintaining visual content of the original images.

1.2 State of the Art and Motivation

For realizing a good E2EE-PHM system, perceptual hashing and its underlying visual feature representation are crucial.

In modern platforms of instant messaging, the widely used perceptual hashing algorithms include pHash [23], Facebook PDQ [11], Microsoft PhotoDNA [32], and Apple NeuralHash [2]. Despite different designs, such algorithms follow a unified workflow: important features reflecting visual content are extracted and then processed (mainly quantized) to form the compact code. Therefore, we can categorize them from the perspective of multiscale feature analysis.

- pHash and PDQ extract *global-scale features* by two-dimensional discrete cosine transform (DCT), with hash codes of 64 and 256 bits, respectively.
- PhotoDNA extract *mid-scale features* by dividing the image into a 6×6 grid and computing gradients within each grid, with hash code of 144 bits.
- NeuralHash extract *pixel-scale features* by convolutional neural network, with hash code of 96 bits.

Motivation. Although the above algorithms have been built into worldwide platforms of instant messaging, their *robustness* is not yet fully understood. Here, adversaries may exploit the robustness weaknesses, thereby fooling E2EE-PHM systems and beyond. More recently, the security community has explored the theoretical feasibility of *adversarial attacks* on E2EE-PHM systems [17, 38]. Practically, such attacks currently still lack real-world threat – only applicable to individual cases or too computationally expensive. For example, Prokos [38] et al. take about 4 GPU hours to generate a successful attack for PhotoDNA. We attribute such limitations to *the mismatch in the scale* between the attack acting features and the hash extracting features. Here, most existing attacks add perturbations only in the pixel scale [8, 14, 31, 45], ignoring the multi-scale nature of global-scale pHash and PDQ, mid-scale PhotoDNA, and pixel-scale NeuralHash.

1.3 Our contribution

Motivated by the above problem, we systematically reveal the adversarial threat to E2EE-PHM systems, leading to regulatory failures. As shown in Figure 1, our attack is a more realistic threat – *uniformly fooling* the famous pHash, Facebook PDQ, Microsoft PhotoDNA, and Apple NeuralHash, even with *higher success rates* and *less training times*. We also validate the above proposition in both scenarios of *escaping* and *triggering* regulation.

- At the **technical** level, we propose a new idea of *multiresolution perturbation*. With its formalized design, each perturbation element can affect image regions of adjustable scales – from the pixel scale to the global scale, like the microscope to the telescope – hence the name ATKSCOPES. Note that such flexible ATKSCOPES encapsulate previous pixel-scale-only attacks as special cases, thus enabling the unified attack across hash algorithms. Moreover, our ATKSCOPES exhibit a huge leap in efficiency, compared to previous ones with mismatched scales between the attack acting features and the hash extracting features.
- At the **practical** level, we achieve uniform, fast, and successful adversarial attacks as realistic threats to 4 commercial hashing algorithms of pHash, PDQ, PhotoDNA, and NeuralHash in 2 scenarios of escaping and triggering regulation. Such practical results reveal the vulnerability of worldwide E2EE-PHM platforms, with potentially serious social implications. From our technical insights, we also discuss possible countermeasures and recommendations for social good.

2 Related Work

In this section, we will introduce the workflow of perceptual hashing algorithms, classify the proposed E2EE content matching systems, and review relevant research on evaluating the robustness of perceptual hashing algorithms.

2.1 Perceptual Hashing

A perceptual hashing algorithm generates a fixed-length data fingerprint (hash code) for multimedia files (e.g., images). Semantically similar images are mapped to similar hash codes. We denote perceptual hashing as a function $\mathcal{H} : \mathbb{I} \to \mathbb{M}^l$. For an input image $X \in \mathbb{I}$, $\mathcal{H}(X) = h$, where $h \in \mathbb{M}^l$, and $\mathbb{M}^l = \{0,1\}^l$ or $\mathbb{M}^l = \mathbb{R}^l$. Here, $\{0,1\}^l$ represents a binary string of length l, and $\mathbb{R}^l$ represents a string of real numbers of length l. An input image X such that $\mathcal{H}(X) = h$ is called a *preimage* of h. Define X' as a slightly modified version of X, where X' and X are visually similar. Also, define Y as an image that is visually different from X. Perceptual hashing should satisfy the following conditions:

- Generate similar hash codes for perceptually similar images. $P(\mathcal{H}(X) = \mathcal{H}(X')) \approx 1$.
- Generate significantly different hash codes for perceptually dissimilar images. $P(\mathcal{H}(X) = \mathcal{H}(Y)) \approx 0$.

The similarity between two images X and Y is quantified by measuring the *distance* between their hashes. Here, we define this distance as $\mathcal{D}$. For binary hashes, $\mathcal{D}$ is calculated using the Hamming distance. For real-number hashes, the distance is calculated using the Euclidean distance.

The process of generating a hash code typically involves three steps. First, the input image X is preprocessed, such as through scaling and grayscaling, to obtain the preprocessed image X^p. Next, the preprocessed image X^p undergoes feature extraction to obtain a feature vector F. Finally, the feature vector F is mapped to a hash code h.

Different hashing algorithms vary in how they extract features. Some algorithms extract local features, such as `NeuralHash` which extracts features in the pixel domain. Some algorithms extract global features, such as `pHash` which converts the image to the frequency domain and then extracts frequency domain features.

2.2 Regulation Under E2EE Privacy

Sensitive images detection via client-side. A client-side matching system based on perceptual hashing for detecting sensitive images includes a database $\mathbb{D} = \{X_1, \ldots, X_N\}$ containing N sensitive images stored on the client side, where each image $X_i \in \mathbb{I}(1 \le i \le N)$, a perceptual hashing algorithm $\mathcal{H}$, a distance measure $\mathcal{D}$ for assessing similarity between hash codes of two images, and a threshold $\Delta_d > 0$ for determining if two images are similar. For an image X uploaded by the user, the system first computes its hash code $\mathcal{H}(X)$ using a perceptual hashing algorithm, and then calculates the distance between this hash code and the hash codes of N images in the database using a function $\mathcal{D}$. If there exists $X_i \in \mathbb{I}(1 \le i \le N)$ such that $\mathcal{D}(\mathcal{H}(X_i), \mathcal{H}(X)) < \Delta_d$, the image is flagged as illicit and further actions are taken. A drawback of this method is deploying the sensitive image database on the client-side, which can be susceptible to reverse engineering. Currently, only Xiaomi [43] employs this method for keyword-based scanning.

Matching with a server-held list of sensitive images. This method stores the sensitive image database $\mathbb{D}$ on the server. Techniques such as Microsoft's EdgeHash [33] method are examples of this approach. The same as the client-side scanning method, this method also uses perceptual hashing algorithm on the client to calculate the hash code of the user content. Assume there is a service provider S that offers an API for client C to use. The client C uses this API to compute the digests EdgeHash(X) of the user-uploaded image X. Then, client C transmits the digests EdgeHash(X) to the centralized provider S for further processing and comparison. The comparison method is similar to that of client-side matching. The key point here is that client C does not need to store the digests of sensitive content locally but directly transmits the computed digests to provider S, who handles subsequent sensitive content matching and processing. The drawback of this method is that it requires transmitting the digests of all client content to the service provider. If an attacker gains access to the client's digests and manages to recover images from the digests, it could result in a privacy breach for users.

Cryptographic matching. To address the shortcomings of the aforementioned methods, recent work [5, 12, 26, 44] has designed privacy-preserving cryptographic protocols for hash matching. In these systems, the content of both the user and the sensitive database is protected from leakage. First, the user content digest $\mathcal{H}(X)$ is computed at the endpoint, and then compared with the provider's sensitive image content database digests through a two-party computation (2PC) protocol. The comparison process can be based on homomorphic encryption [44] or private set intersection (PSI) [5] protocols.

2.3 Robustness of Perceptual Hashing

Previous research has shown that perceptual hashing algorithms are robust against common image transformations such as resizing, recoloring, watermarking, cropping, and blurring [1, 10, 55]. These studies explored the performance of perceptual hashing in duplicate image detection. However, they did not further consider the impact of adversarial attacks on the robustness of perceptual hashing.

A recent study [8] investigate achieving hash collisions through adversarial attacks. Perceptual hashing algorithms are also widely used for similar image search and matching on the internet, and many researchers have evaluated their robustness in image retrieval [4, 14, 15, 27, 28, 47, 48, 53, 54].

During our investigation, several concurrent projects [17, 38, 45] also examine the security of perceptual hashing functions in the context of E2EE. In 2021, Apple release a new system called `NeuralHash` [2], which focuses on identifying Child Sexual Abuse Material (CSAM) content uploaded by users to Apple's iCloud service. Subsequently,

researchers [45] conduct a comprehensive analysis of the NeuralHash-based deep perceptual hashing content matching system, demonstrating that attackers could fine-tune images by applying gradient-based methods or simple image transformations to change the image's hash code, forcing hash collisions or evading detection. Although this is related to the attack scenarios discussed in this paper, NeuralHash, as a deep perceptual hashing algorithm, is inherently susceptible to adversarial attacks. However, the non-learning perceptual hashing algorithms evaluated in this paper (e.g., PhotoDNA, PDQ, and pHash) require a more complex optimization framework.

Jain et al. [17] evaluate the robustness of client-side matching systems based on perceptual hashing against detection avoidance attacks. They evaluate shallow hashing algorithms such as pHash, aHash, dHash, and PDQ but lack an evaluation of deep hashing and only consider a single attack scenario focused on detection avoidance.

Prokos et al. [38] refine the work of [17] by considering two attack scenarios, including escaping regulation attacks and triggering regulation attack. They develop a threat model for perceptual hashing algorithms and evaluate the two most widely deployed non-learning perceptual hashing algorithms: PhotoDNA and PDQ. However, their optimization process only considered adding perturbations in the image's pixel domain, ignoring the different resolutions that the hashing algorithms might involve when extracting features, leading to prolonged attack times. An attack on PhotoDNA takes about 4 hours to complete 20,000 training rounds for a single triggering regulation attack. In real-world attack scenarios, this undoubtedly requires significant time and computational resources.

3 Attack Modeling

We here propose two threat models against E2EE-PHM systems, referred to as the escaping regulation attack and the triggering regulation attack.

3.1 Escaping Regulation Attack

We assume that the attacker has access to an image from the sensitive content database $\mathbb{D}$, denoted as the original image $X \in \mathbb{D}$.

The attacker's objective is to circumvent the sensitive image database $\mathbb{D}$ and distribute this image across the network within the framework of the E2EE-PHM systems. In details, the attacker seeks to create an attack image X' that fulfills two conditions:

- The original image X and the attack image X' remain visually similar.
- The attack image X' escapes regulation, meaning $\mathcal{D}(\mathcal{H}(X'), h_X) > \Delta_d$, where h_X represents the hash code of the original image and Δ_d denotes the matching threshold of the E2EE-PHM system.

Here, we assume the attacker knows Δ_d. We will discuss the validity of this assumption in Section 5.2.

Formulation 1. ***(Escaping Regulation Attack: Optimization Objective).*** *With the above two conditions, the objective of the escaping regulation attack can be formalized as follows:*

$$\begin{aligned} \textit{Minimize: } & \mathcal{V}(X, X') + c \cdot \mathcal{L}(X', h_X) \\ \textit{s.t.: } & \mathcal{D}(\mathcal{H}(X'), h_X) > \Delta_d \\ & X' \in \mathbb{I} \end{aligned} \quad , \tag{1}$$

where $\mathcal{V}$ denotes the measure of visual similarity between two images, with a lower value indicating higher visual similarity. The function $\mathcal{L}$ represents a differentiable loss function that allows the optimization process to gradually increase the distance between the hash codes of images X and X' until it satisfies $\mathcal{D}(\mathcal{H}(X'), h_X) > \Delta_d$. The parameter c is a adjustable weight factor employed to balance the two loss terms. The function $\mathcal{D}$ quantifies the distance between the hash codes of images X and X':

$$\mathcal{D} = \begin{cases} \mathcal{D}_{\mathbb{B}} & \textit{if the hash code is binary} \\ \mathcal{D}_{\mathbb{R}} & \textit{otherwise} \end{cases},$$

where when the hash codes are binary, the Hamming distance, denoted as $\mathcal{D}_{\mathbb{B}}$, is used to quantify the number of differing bits between the hash codes. Otherwise, the L_1 distance, denoted as $\mathcal{D}_{\mathbb{R}}$, is used to calculate the distance between the hash codes.

The attacker's goal is to minimize the function $\mathcal{L}$ while preserving the content of the image as much as possible. The first constraint requires that the distance between the hash codes of the original image X and the attack image X' must exceed the threshold Δ_d, thereby preventing the E2EE-PHM systems from matching the attack image to the original. The second constraint mandates that the attack image X' must be valid, with pixel values within the allowable range.

If the attacker can achieve the above objective at a practically feasible cost, they can effectively distribute any sensitive image without being detected.

3.2 Triggering Regulation Attack

In this threat model, the attacker aims to create a pair of semantically unrelated images that collide in the hash space.

Specifically, the attacker adds perturbations to an innocuous image X to generate an attack image X', such that the hash code of X' matches the hash code h_t of a target sensitive image in the database. X and X' should appear visually similar.

In practice, the attacker might send X' via a communication medium that does not reveal the recipient's identity (e.g.,

bulletin boards or anonymous messaging systems), and the recipient might forward X' through the same medium. Any user forwarding X' through E2EE-PHM systems would be falsely detected as transmitting sensitive content. Given the distribution across many devices, this could lead to a large number of false alarms in E2EE-PHM systems, resulting in the entrapment of innocent users or a Distributed Denial of Service (DDoS) attack.

Another concerning scenario is that governments or other powerful organizations might exploit this method to manipulate images for censorship purposes. For instance, they could manipulate images critical of the government to collide with hash codes in the sensitive database and then disseminate these manipulated images via social media. Ultimately, these images may appear on the devices of supporters of such rhetoric, at which point the E2EE-PHM systems would trigger a match, thus facilitating the identification, monitoring, and persecution of these individuals.

We assume that the attacker knows the hash code h_t of a target image in the sensitive database. There is no need to know the original image corresponding to the hash code h_t. For a benign input image X, the attacker aims to generate an attack image X' that meets the following two conditions:

- The original image X and the attack image X' remain visually similar.
- The attack image X' triggers regulation, meaning $\mathcal{D}(\mathcal{H}(X'), h_t) < \Delta_d$, where Δ_d represents the matching threshold of the detection system.

Here, we assume the attacker knows Δ_d.

Formulation 2. ***(Triggering Regulation Attack: Optimization Objective).*** *With the above two conditions, the objective of the triggering regulation attack can be formalized as follows:*

$$\begin{aligned} \underset{\delta}{\text{Minimize:}} & \mathcal{V}\left(X', X\right) + c \cdot \mathcal{L}(X', h_t) \\ s.t.: & \mathcal{D}(\mathcal{H}(X'), h_t) < \Delta_d \\ & X' \in \mathbb{I} \end{aligned} \quad , \tag{2}$$

where $\mathcal{V}$ and $\mathcal{D}$ are consistent with the definitions provided in Formulation 1. The function $\mathcal{L}$ represents a differentiable function that enables the hash code of the attack image X' generated during the optimization process to gradually approach the hash code of the target image h_t until it satisfies $\mathcal{D}(\mathcal{H}(X'), h_t) < \Delta_d$. The parameter c serves as a tunable weight factor to balance the two loss terms.

The attacker's goal is to minimize the function $\mathcal{L}$ while ensuring that the adversarial image X' retains the visual content of the original image X as much as possible. The first constraint ensures that the distance between the hash code of the attack image X' and the target hash code h_t is less than the detection system's matching threshold Δ_d. This allows the attack image X' to be mistakenly recognized by the detection system as the target image. The second constraint requires that the attack image X' must remain within the valid pixel range.

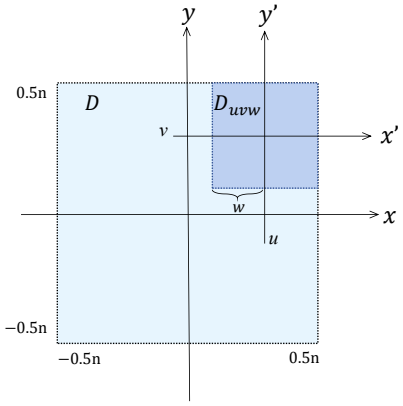

Figure 2: Illustration of local orthogonal transformation, where D represents the image domain and D_{uvw} represents the domain of the basis function, converting the two domains with the translation offset (u, v) and the scaling factor w.

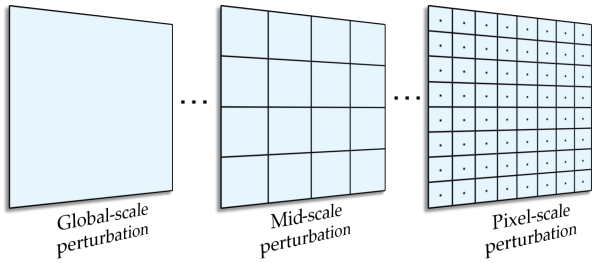

Figure 3: Illustration of multiresolution perturbation.

4 Attack Methodology

In this section, we discuss the methodology of ATKSCOPES for performing attacks against `pHash`, `PhotoDNA`, `PDQ`, and `NeuralHash`.

4.1 Escaping Regulation Attack

We define a valid image $\mathbb{I} = [0,1]^n$, where the image size n equals the number of pixels in the image. Given an input image X and a perceptual hash function $\mathcal{H}$, the goal is to find the minimal perturbation such that the attack image X' of the original image X is not matched to X during content matching. This corresponds to finding an X' semantically equivalent to X, where $\mathcal{D}(\mathcal{H}(X'), h_X) > \Delta_d$. Here, $\mathcal{D}$ represents the distance between the hash codes of the two images and h_X represents the hash code of X. When the hash distance

between the two images exceeds the detection threshold of E2EE-PHM, it will not trigger a hash match under the given threshold. Our optimization objective is consistent with that described in Formulation 1.

4.1.1 Multiresolution Perturbation

Unlike previous studies that directly add perturbations to the pixel domain of the image (e.g., $X+\delta$), we introduce multiresolution perturbation. Here, each perturbation element can affect image regions of adjustable scales, for a good match in scales between the attack acting features and the hash extracting features, thereby significantly changing the hash codes.

Definition 1. ***(Multiresolution perturbation).*** *The addition of multiresolution perturbation is defined as follows:*

$$X'_{(x,y)\in D_{uvw}} = \mathcal{F}^{-1}\left(\mathcal{F}(X)+\delta\right), \tag{3}$$

with notations of

$$\mathcal{F}(X) = \langle X, V_{nm}^{uvw}\rangle = \iint_D (V_{nm}^{uvw}(x,y))^* X(x,y)dxdy, \tag{4}$$

and

$$\mathcal{F}^{-1}(\mathcal{F}(X)) = \sum_{n,m} V_{nm}^{uvw}(x,y)\mathcal{F}(X), \tag{5}$$

where $\mathcal{F}$ denotes the local orthogonal transformation [39], with image $X(x,y)$ on domain $(x,y)\in D$. The local orthogonal basis function V_{nm}^{uvw} is defined on the domain D_{uvw} with the order parameters $(n,m)\in\mathbb{Z}^2$, converting D to D_{uvw} by the translation offset (u,v) and the scaling factor w, as illustrated in Figure 2. Note that the local orthogonal basis function V_{nm}^{uvw} can be defined from any global orthogonal basis function V_{nm}, e.g., a family of harmonic functions, with following form:

$$V_{nm}^{uvw}(x,y) = V_{nm}(x',y') = V_{nm}\left(\frac{x-u}{w}, \frac{y-v}{w}\right). \tag{6}$$

Remark. Multiresolution perturbations are in fact a very general formalization, covering previous pixel-scale-only attacks as special cases. As shown in Figure 3, such flexible ATKSCOPES allow for a unified attack across hash algorithms, from global-scale `pHash` and `PDQ`, to mid-scale `PhotoDNA`, and to pixel-scale `NeuralHash`. Next, we will further explain such typical examples derived from Definition 1.

We define the offsets $u\in\mathbb{U}$ and $v\in\mathbb{V}$, the scaling factor $w\in\mathbb{W}$. The combined set of all possible parameters is denoted as $\Omega=\mathbb{U}\times\mathbb{V}\times\mathbb{W}$. Technically, we first transform the image X to obtain its coefficient matrix $\mathcal{F}(X)$. Then, we add perturbation δ into the coefficient matrix $\mathcal{F}(X)$. For the perturbed image X', we perform an inverse transformation $\mathcal{F}^{-1}$ on the perturbed matrix and sum the inverse results over all $(u,v,w)\in\Omega$.

Example 1. ***(Global-scale perturbations).*** *For global-scale perturbations, the offsets are defined as $\mathbb{U}=0$ and $\mathbb{V}=0$ with scaling factor of $\mathbb{W}=1$*

Global-scale perturbations are good matches for the perceptual hash algorithms that rely on the global features of the image, such as `pHash` and `PDQ`.

Example 2. ***(Pixel-scale perturbations).*** *For pixel-scale perturbations, the offsets are defined as $\mathbb{U}=\{-n+1,-n+3,\ldots,n-1\}/2$ and $\mathbb{V}=\{-n+1,-n+3,\ldots,n-1\}/2$ with the scaling factor of $\mathbb{W}=2/n$, where n is the scale of the image.*

Pixel-scale perturbations are good matches for the perceptual hash algorithms that rely on the pixel features of the image, such as `NeuralHash`. Here, traditional pixel-domain perturbations [14, 17, 29, 38] are special cases of multiresolution perturbations.

Example 3. ***(Mid-scale perturbations).*** *For mid-scale perturbations, the offsets are defined as $\mathbb{U}=\{-n+2k,-n+6k,\ldots,n-2k\}/2$ and $\mathbb{V}=\{-n+2k,-n+6k,\ldots,n-2k\}/2$, with the scaling factor of $\mathbb{W}=2k/n$, where k is a factor of n.*

Mid-scale perturbations are good matches for the perceptual hash algorithms that rely on the patch features of the image, such as `PhotoDNA`.

4.1.2 Visual Distance $\mathcal{V}$

The visual distance $\mathcal{V}$ is used to measure the visual similarity between two images. In this paper, we consider two visual metrics, including L_2 distance and Learned Perceptual Image Patch Similarity distance (LPIPS) [57].

L_2 distance, also known as Euclidean distance, is a direct measurement method that calculates the square root of the sum of the squared differences between the corresponding pixel values of two images. This metric is computationally efficient and widely used in various image processing tasks.

LPIPS distance is a learned metric method that compares images based on features extracted by deep neural networks. LPIPS aims to better align with human perceptual judgment by capturing subtle differences in texture, color, and structure.

In our attacks on four perceptual hashing algorithms, we use these two visual metrics as loss terms to evaluate the impact of different visual distances on the optimization process.

4.1.3 Hash Similarity Loss $\mathcal{L}$

Recall that $\mathcal{L}$ needs to be differentiable and reflects the similarity of the hash codes of two images. We optimize $\mathcal{L}$ to increase the distance between semantically similar images in the hash space, thereby escaping regulation.

Definition 2. *($\mathcal{L}$ for `pHash`, `PDQ`, and `PhotoDNA`). To change the hash code of the input image X such that the hash distance between h_X and $\mathcal{H}(X')$ exceeds a given threshold Δ_d, a hinge loss function is defined as follows:*

$$\mathcal{L}(X', h_X) = \max\left\{\tanh\left(1 - \frac{\mathcal{D}(\mathcal{H}(X'), h_X)}{\Delta d}\right), 0\right\}, \quad (7)$$

where $\mathcal{L}$ converts the distance between the two hash strings into a probability. Δ_d is a tunable parameter that represents the desired distance in the hash space between the attack image and the original image.

Remark. For instance, for a 256-bit hash output from `PDQ`, if an attacker aims to create an attack image X' with an 80-bit difference in the hash space from the original image X, Δ_d should be set to 80. We use tanh() for $\mathcal{L}$ because it makes the loss function steeper as the hash distance approaches the target distance, thereby facilitating the convergence of the optimization process.

Definition 3. *($\mathcal{L}$ for `NeuralHash`). For `NeuralHash`, gradient-based white-box optimization is employed to increase the hash difference between the original image X and the attack image X'. The differentiable $\mathcal{L}$ is defined as follows:*

$$\mathcal{L}(X', h_X) = -L_1(\mathcal{S}(p),\ h_X), \quad (8)$$

where $p = \mathcal{M}(X')$ represents the floating-point hash output vector of length 96 before binarization. $\mathcal{M}$ signifies the feature extraction and transformation process in the `NeuralHash` model. Specifically, $\mathcal{M}$ maps the input image X' to a floating-point vector p of length 96. In `NeuralHash`, the Heaviside step function is used to binarize p into a binary vector. To make the loss function differentiable, the binarization step is replaced with a Sigmoid function $\mathcal{S}(p) = \frac{1}{1+\mathbf{e}^{-p}}$, thereby allowing gradients to flow through the network during optimization. Instead of L_1, $-L_1$ is increasing the Hamming distance between two hashes.

4.1.4 Optimization Process

Gradient-based optimization. Algorithm 1 describes our optimization process. The original image X is used as the starting image for the attack. Our goal is to find an attack image X' such that $\mathcal{D}(\mathcal{H}(X'), h_X) > \Delta_d$. We generate the attack image X' by adding perturbations to the original image X, where the magnitude of the perturbations is determined by

Algorithm 1 Escaping regulation attack

Inputs: N: num training rounds, X: original image to be attacked, h_X: hash code of original image, $f()$: loss function, Δd: E2EE-PHM systems matching threshold, (u, v): translation offset, w: scale factor, γ: learning rate.
Output: X': attack image
for $i = 1, 2, \ldots, N$ **do**
  $\delta_i = \texttt{calcDelta}(\delta_{i-1}, f, \gamma)$
  $X_i \leftarrow \sum_{u,v,w \in \Omega} \mathcal{F}^{-1}(\mathcal{F}(X_{i-1}) + \delta_{i,uvw})$
  $HashDistance = \mathcal{D}(\mathcal{H}(X_i), h_X)$
  if $HashDistance > \Delta_d$ **then**
    successfully generated attack image
  end if
end for

Algorithm 2 calcDelta

Inputs: $f()$: loss function, γ: learning rate, δ_i: perturbation, $M \in \mathbb{R}^q$, $\upsilon \in \mathbb{R}^q$, $T \in \mathbb{Z}^q$: ADAM states, where q is the size of the perturbation, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$: ADAM hyper-parameters, a: a small constant, $\mathbf{e}_i$: a standard basis vector with only the i-th component as 1.
Output: δ_{i+1}
randomly pick a coordinate $i \in \{1, \cdots, q\}$
if attack mode = black-box **then**
  $\hat{g}_i := \frac{\partial f(\delta)}{\partial \delta_i} \approx \frac{f(\delta + a\mathbf{e}_i) - f(\delta - a\mathbf{e}_i)}{2a}$
else
  $\hat{g}_i := \frac{\partial f(\delta)}{\partial \delta_i}$ using gradient propagation
end if
$T_i \leftarrow T_i + 1$
$M_i \leftarrow \beta_1 M_i + (1 - \beta_1)\hat{g}_i$
$v_i \leftarrow \beta_2 v_i + (1 - \beta_2)\hat{g}_i^2$
$\hat{M}_i = M_i / (1 - \beta_1^{T_i})$
$\hat{v}_i = v_i / (1 - \beta_2^{T_i})$
$\delta_{i+1} = -\gamma \frac{\hat{M}_i}{\sqrt{\hat{v}_i} + \varepsilon}$

Algorithm 2. Initially, we perform an image domain transformation of the image X in the local coordinate system (u, v) with a scaling factor w, obtaining its coefficient matrix representation $\mathcal{F}(X)$ in the transformed domain. The perturbation is then applied to the transformed image domain as $\mathcal{F}(X_{i-1}) + \delta_{i,uvw}$. Subsequently, we obtain the updated local image through an inverse image transformation. Finally, by stitching together the inverse-transformed local images for all $(u, v, w) \in \Omega$, we reconstruct the complete image X' with added perturbations. This process is repeated until the hash distance between the updated image and the original image satisfies $\mathcal{D}(\mathcal{H}(X_i), h_X) > \Delta_d$.

Perturbation update. Algorithm 2 details the perturbation update. In each training round, a coordinate of the pertur-

bation variable is randomly selected, and it is updated by approximately minimizing the objective function along that coordinate. For black-box attacks on non-learning perceptual hashing algorithm, we estimate the gradient using the symmetric difference quotient. In detail, we estimate the gradient by adding and subtracting a small constant a to the perturbation δ, and then computing the difference in the loss function values. For white-box attacks on `NeuralHash`, we use backpropagation to estimate the gradient. After obtaining the gradient values, we update the selected coordinate using ADAM, and compute and return the updated perturbation δ_{i+1}.

4.2 Triggering Regulation Attack

Given an input original image X and the hash code h_t of a target image in an sensitive image database, we do not even need to know the source image corresponding to h_t. Our goal is to find a minimal perturbation δ such that X' remains semantically similar to the original image X, but $\mathcal{D}(\mathcal{H}(X'),h_t) < \Delta_d$. This would cause the distance between two semantically different images in the hash space to be below the detection threshold of the E2EE-PHM system, leading to numerous false alarms within the system. Our optimization objective is as described in Formulation 2.

4.2.1 Visual Distance $\mathcal{V}$

In this attack scenario, we also consider two visual similarity measures, including L_2 distance and LPIPS. We use these metrics as separate loss functions $\mathcal{V}$ and test their performance in attacks against four different perceptual hash algorithms. Detailed results will be presented in the experiments section 5.3.2.

4.2.2 Hash Similarity Loss $\mathcal{L}$

The loss function $\mathcal{L}$ needs to be differentiable and reflect the similarity between the hash codes of two images. This ensures that, when minimizing the value of the loss function $\mathcal{L}$, semantically different images can collide in the hash space.

Definition 4. *($\mathcal{L}$ for `pHash`, `PDQ`, and `PhotoDNA`). A hinge loss function $\mathcal{L}$ for `pHash`, `PDQ` and `PhotoDNA` is defined as follows:*

$$\mathcal{L}(X',h_t) = max\{\mathcal{D}(\mathcal{H}(X'),h_t),0\}, \tag{9}$$

Definition 5. *($\mathcal{L}$ for `NeuralHash`). A hinge loss function $\mathcal{L}$ for `NeuralHash` is defined as follows:*

$$\mathcal{L}(X',h_t) = max\{0,-p\cdot\psi(h_t)\}, \tag{10}$$

where $p = \mathcal{M}(X')$ is the floating-point hash output before binarization. $\mathcal{M}$ maps the input image X' to a floating-point vector p of length 96. The operation $\psi(h_t) = \text{sign}(h_t - 0.5)$ transforms each 0-bit in the hash vector with -1. The hash code $\mathcal{H}(X')$ equals h_t if and only if the signs of p and $\psi(h_t)$ match at every bit of the hash code.

Algorithm 3 Triggering regulation attack

Inputs: N: num training rounds, X: original image to be attacked, h_t: hash code of target image, $f()$: loss function, Δd: E2EE-PHM systems matching threshold, (u,v): translation offset, w: scale factor, γ: learning rate.
Output: X': attack image
for $i = 1,2,\ldots,N$ **do**
  $\delta_i = \texttt{calcDelta}(\delta_{i-1},f,\gamma)$
  $X_i \leftarrow \sum_{u,v,w\in\Omega} \mathcal{F}^{-1}(\mathcal{F}(X_{i-1}) + \delta_{i,uvw})$
  $HashDistance = \mathcal{D}(\mathcal{H}(X_i),h_t)$
  if $HashDistance < \Delta_d$ **then**
    successfully generated attack image
  end if
end for

4.2.3 Optimization Process

Algorithm 3 describes the process of Triggering Regulation Attack. The original image X is input as the starting image for optimization, and the hash code h_t is input as the target hash code. Our goal is to find an adversarial image X' that satisfies $\mathcal{D}(\mathcal{H}(X'),h_t) < \Delta_d$. The function `calcDelta` is used to compute the perturbation δ_i, which employs a loss function f and a learning rate γ to determine the direction and magnitude of the perturbation. The process of adding perturbation to the image is identical to Escaping Regulation Attack. During the iterative process, $\mathcal{D}(\mathcal{H}(X_i),h_t)$ is continuously checked to see if it is less than the threshold Δ_d. If the distance is below this threshold Δ_d, the attack is considered successful, and the algorithm terminates early.

5 Evaluation

5.1 Experimental Setup

Perceptual hashing algorithms. We select an open-source implementation [7] of pHash, which has been utilized in many prior studies [14, 21, 22, 37, 40]. We set `pHash` with there recommended parameters. For the implementation of `PDQ`, we use the official source code released by Facebook [11]. `PhotoDNA` is proposed by Microsoft, but Microsoft has not provided any open implementation of `PhotoDNA` to date. However, a presumed implementation [19, 20] of `PhotoDNA` was leaked on GitHub in 2021 , and we use it for our attacks. Its behavior is similar to the high-level description of `PhotoDNA` provided by one of its algorithm authors [12]. For `NeuralHash`, we extract the `NeuralHash` model from recent versions of macOS or iOS.

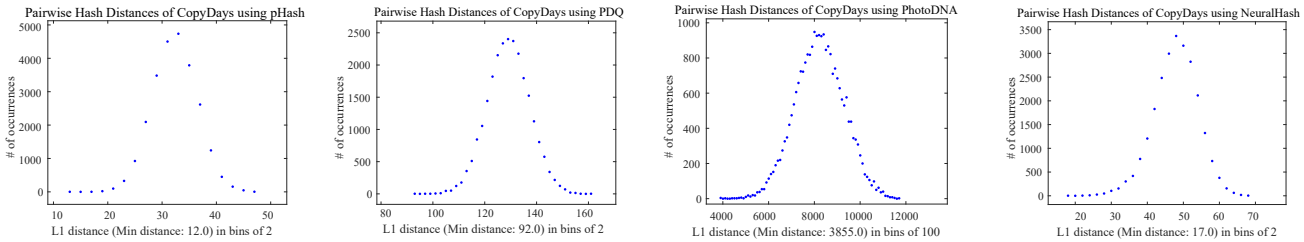

Figure 4: Frequency distribution of paired hash distances derived from 12246 perceptually different image pairs in the CopyDays dataset. These distributions are used to determine the threshold for image detection. From left to right, the graphs represent the frequency distributions for `pHash`, `PDQ`, `PhotoDNA` and `NeuralHash`.

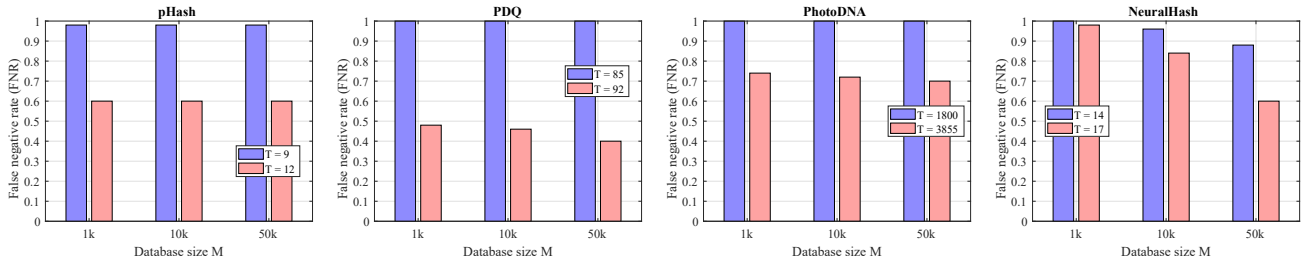

Figure 5: FNR for attack images using `pHash`, `PDQ`, `PhotoDNA`, and `NeuralHash` with different scales of database $\mathbb{D}$.

Datasets. For our evaluation, we use two different image databases, which will be used for different purposes. We evaluate our two attack scenarios using the ImageNet dataset from the ILSVRC 2012 challenge [41]. The ImageNet dataset is characterized by its large scale and diversity. We assume that every pair of images in the ImageNet dataset is visually distinct. We randomly select 50 pairs of images from the ImageNet dataset for the triggering regulation attack. For the escaping regulation attack , we also randomly select 50 images from the ImageNet dataset as the start images.

We use the CopyDays dataset [16] to calculate the detection thresholds for each perceptual hashing algorithm because the hash distances between image pairs are relatively similar and stable across different datasets [17], allowing attackers to reasonably estimate the detection thresholds.

Hardware Specifications. All experiments in this section were compiled and executed using PyTorch 1.9 and Python 3.6 on two separate machines. PhotoDNA attacks were conducted on a 6-Core 3.0-GHz CPU (Intel Core i5-12490F). PDQ and pHash attacks were executed on a 12-Core 3.6-GHz CPU (Intel Core i7-12700K), while NeuralHash attacks were performed on a 14592-Core 2.3-GHz GPU (Nvidia RTX 4090).

5.2 Baseline Experiments

Determining match thresholds. Before running the attacks, we need to determine the appropriate distance threshold Δd for image matching with different perceptual hashing algorithms. The threshold Δd set in E2EE-PHM systems has a

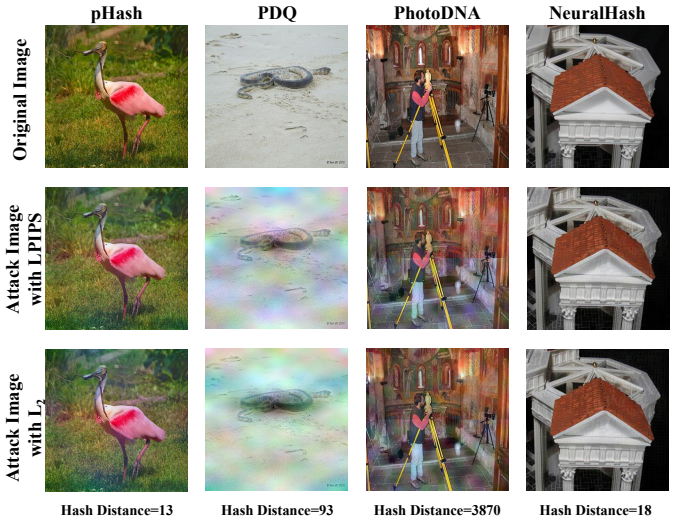

Figure 6: Example images for escaping regulation attack against `pHash`, `PDQ`, `PhotoDNA` and `NeuralHash`.

complex impact on the matching results. A higher threshold will make the system more sensitive to attack images generated by escaping regulation attack but will also result in a higher rate of natural false positives. A lower threshold will make triggering regulation attack more difficult to achieve but will increase the success rate of escaping regulation attack.

We calculate the hash distances between each pair of images in the CopyDays dataset, which consists of 157 perceptually distinct images, resulting in a total of $(157 \times 156)/2 = 12246$ image pairs. The hash distance frequency distribution

Success Rate	L_2 Distance ↓	Hash Distance	Rounds	k
100%	104.29	3851.02	479.12	$n/2$
100%	77.52	3850.92	551.60	$n/4$
100%	92.84	3851.01	756.01	$n/8$

Table 1: Comparison of the effects of different k values in the triggering regulation attack on PhotoDNA. Where $k = n/4$ and $k = n/8$ indicate that the domain of the basis function is divided into 4 equal parts and 16 equal parts of the image, respectively. When $k = n/2$, it means that the domain of the basis function is the same as the domain of the image. Here, we use the cosine function as the basis function, corresponding to the discrete cosine transform.

$\mathcal{H}$	$\mathcal{V}$	Success Rate	L_2 Distance ↓	LPIPS Distance ↓	Hash Distance	Δ_d	Rounds
pHash	L_2	100%	66.43	0.22	13.00	12	122.36
pHash	LPIPS	100%	79.51	0.24	13.00	12	118.52
PDQ	L_2	100%	73.03	0.49	92.52	92	79.36
PDQ	LPIPS	100%	96.84	0.45	92.64	92	69.86
PhotoDNA	L_2	100%	46.82	0.23	3866.04	3855	137.64
PhotoDNA	LPIPS	100%	59.88	0.21	3858.00	3855	132.22
NeuralHash	L_2	100%	30.62	0.19	18.32	17	18.66
NeuralHash	LPIPS	100%	30.78	0.19	18.36	17	18.84

Table 2: Escaping regulation attack against pHash, PDQ, PhotoDNA and NeuralHash with different perceptual distance functions $\mathcal{V}$.

results for 12246 image pairs are shown in Figure 4. From the figure, we find that the appropriate image matching thresholds Δ_d for pHash, PDQ, PhotoDNA, and NeuralHash are 12, 92, 3855, and 17, respectively.

Attack parameters. For different perceptual hashing algorithms, we apply perturbations at various scales using Definition 1. For PDQ and pHash, we apply global-scale perturbations of Example 1. For NeuralHash, we apply local-scale perturbations of Example 2. For PhotoDNA, we apply mid-scale perturbations of Example 3. To determine the appropriate value of k for attacking PhotoDNA, we conducted a series of controlled experiments. These controlled experiments were performed under the triggering regulation attack scenario, using the L_2 distance to measure visual distortion, with the matching threshold Δ_d set to 3855. Table 1 reports the results of different k values in the triggering regulation attack on PhotoDNA, where $k = n/4$ results in the smallest visual loss for the attack images. When there is not much difference in the number of training rounds, we choose parameter settings that can minimize visual loss. In the following experiments, we will use the above parameter settings to attack PhotoDNA.

5.3 Attack Results

We conduct empirical evaluations of the following two attacks using the ImageNet dataset from the ILSVRC 2012 challenge [41].

5.3.1 Escaping Regulation Attack

In our first attack scenario, we investigate the feasibility of generating images that can escape regulation by the E2EE-PHM system. Specifically, we explore whether perturbations applied to images could change their hash codes enough to bypass the E2EE-PHM system's detection of sensitive material. We randomly selecte 50 visually distinct images from the ImageNet dataset as the starting image for our attacks. Based on the baseline experiments described in Section 5.2, we set the matching thresholds Δ_d for pHash, PDQ, PhotoDNA, and NeuralHash to 12, 92, 3855, and 17, respectively.

Impact of perceptual distance function $\mathcal{V}$. We evaluate the impact of using L_2 and LPIPS as the visual metric $\mathcal{V}$, respectively, on the effectiveness of the attack in the escaping regulation attack against four perceptual hashing algorithms. Table 2 presents the final hash distances between the attack images and the original images. Additionally, we report the perceptual differences between the attack images and the original images, measured using L_2 distance (denoted as" L_2 distance") and LPIPS (denoted as "LPIPS distance").

Firstly, table 2 shows that using both L_2 and LPIPS as loss functions $\mathcal{V}$ achieves the intended attack effect. Both methods are able to make $\mathcal{D}(\mathcal{H}(X'), h_X) > \Delta_d$. Secondly, by comparing L_2 and LPIPS, we observe that using L_2 as the loss function results in a smaller L_2 distance between the generated attack image and the original image at each pixel but requires slightly more training rounds to complete the attack. In contrast, using LPIPS as $\mathcal{V}$ creates attack images

$\mathcal{H}$	$\mathcal{V}$	Success Rate	L_2 Distance ↓	LPIPS Distance ↓	Hash Distance	Δ_d	Rounds
pHash	L_2	100%	78.83	0.30	10.61	12	238.94
pHash	LPIPS	100%	85.45	0.29	10.80	12	220.34
PDQ	L_2	100%	76.60	0.46	90.80	92	64.60
PDQ	LPIPS	100%	84.16	0.42	90.80	92	58.60
PhotoDNA	L_2	100%	77.52	0.43	3850.92	3855	551.02
PhotoDNA	LPIPS	100%	102.59	0.48	3851.28	3855	496.21
NeuralHash	L_2	98%	48.60	0.33	13.30	17	762.04
NeuralHash	LPIPS	92%	49.79	0.27	13.13	17	409.45

Table 3: Triggering regulation attack against `pHash`, `PDQ`, `PhotoDNA` and `NeuralHash` with different perceptual distance functions $\mathcal{V}$.

with lower perceptual changes (LPIPS distance is generally smaller when LPIPS is used as $\mathcal{V}$ compared to L_2) and can complete the attack with less training rounds

Results. In our experiments, we successfully execute attacks on 50 randomly selected ImageNet sample images, achieving a 100% attack success rate. Figure 16 illustrates examples of images attacked successfully using our attack, with the hash distances between the attack images and original images all exceeding the predefined threshold Δ_d. The attack images consistently retained the visual content of the original images. Table 2 presents the visual differences between the attack images and the original images, measured using L_2 and LPIPS, with `PDQ` requiring the most significant visual modifications. Figure 6 also shows the attack images generated using LPIPS and L_2 as visual loss constraints, respectively. Additionally, Table 2 indicates that our proposed method achieves the attack with very few training rounds. On average, we exceed the threshold $\Delta d = 3855$ for `PhotoDNA` within 138 training rounds For `PDQ`, we exceed $\Delta d = 92$ in just 80 training rounds. Similarly, for `pHash` and `NeuralHash`, only 123 and 19 training rounds are needed to exceed their evasion detection thresholds of $\Delta d = 12$ and $\Delta d = 17$, respectively.

Real-world attack scenario replication. In the aforementioned attack, we define an attack as successful when the distance between the perturbed image X' and the original image X exceeds the threshold Δd. However, a scenario may arise where the distance between the perturbed image X' and the original image X is greater than Δd, but the distance between X' and another image in the sensitive content database $\mathbb{D}$ is less than or equal to Δd. Consequently, the image is correctly flagged as sensitive content, although for the wrong reason. To evaluate the impact of this scenario on the evasion effectiveness of our escaping regulation attacks against E2EE-PHM systems, we design the following experiment.

First, we assume that the 50 attack images represent images with sensitive content in real-world scenarios. Next, we construct the sensitive content database $\mathbb{D}$ by selecting M images from ImageNet, which includes the 50 original images corresponding to the 50 attack images. For each perceptual

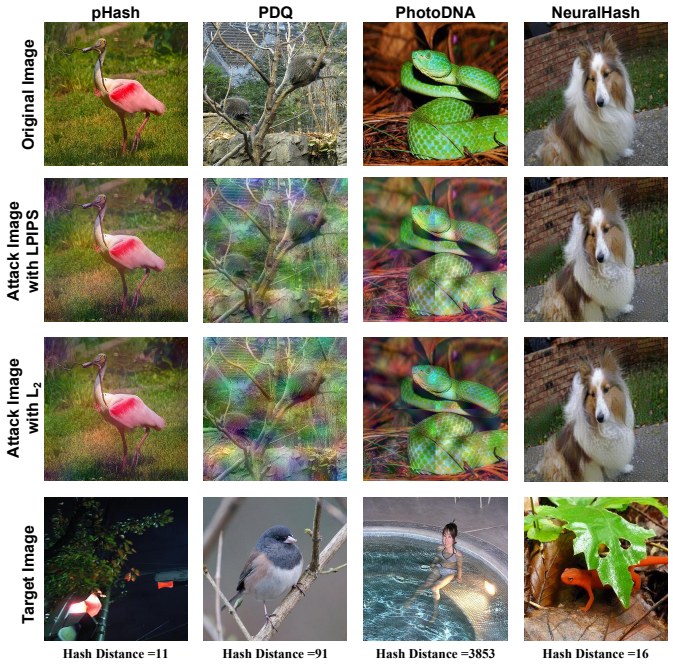

Figure 7: Example images for triggering regulation attack against `pHash`, `PDQ`, `PhotoDNA` and `NeuralHash`.

hashing algorithm, we set two detection thresholds, ensuring that these thresholds do not exceed the values established in the baseline experiment. If the thresholds used in practice exceed those from the baseline experiment, it could result in an excessive number of false positives, incorrectly labeling benign images as sensitive content. This is because the hash distance between two benign images might also be less than or equal to the baseline threshold Δd.

We then compute the hash distance between each attack image and the images in the sensitive content database $\mathbb{D}$. If the distance is less than the given threshold Δd, the image is flagged as sensitive content. Finally, we calculate the proportion of attack images incorrectly rejected by the detection system, which represents the false negative rate (FNR).

As illustrated in Figure 5, for the perceptual hashing algo-

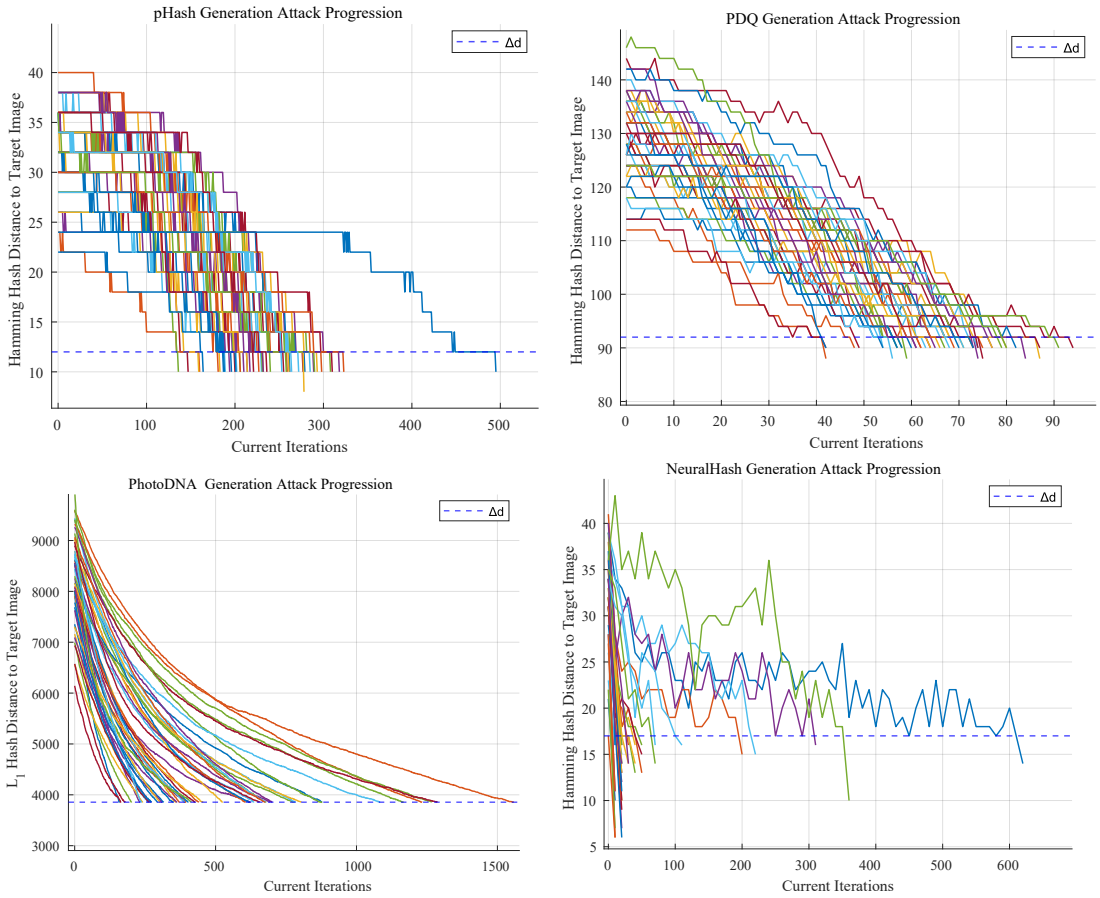

Figure 8: The progress of triggering regulation attack using `pHash`, `PDQ`, `PhotoDNA`, and `NeuralHash` on 50 randomly selected image pairs from the ImageNet dataset. The baseline Δ_d is shown by the blue dashed line. The Δ_d values for `pHash`, `PDQ`, `PhotoDNA`, and `NeuralHash` are 12, 92, 3855, and 17, respectively.

rithms `pHash`, `PDQ`, `PhotoDNA`, and `NeuralHash`, with thresholds set at 9, 85, 1800, and 14 respectively, the false negative rate (FNR) for attack images is exceedingly high across all algorithms. This indicates that very few attack images get flagged for being at a hash distance $< \Delta_d$ from another image in the database. As the size of the database increases, the FNR decreases as expected. With a larger dataset, there is a higher likelihood of being flagged, even if the reason is incorrect.

5.3.2 Triggering Regulation Attack

In this attack scenario, given the hash code h_t of a target image X_t, we attempt to generate an image X' that collides with h_t in the hash space, such that $\mathcal{D}(\mathcal{H}(X'), h_t) < \Delta_d$. This is equivalent to matching benign images with images in a database of sensitive content in the E2EE-PHM system, leading to a high number of false positives in the system.

Impact of Perceptual Distance Function $\mathcal{V}$. Table 3 shows that for non-learning-based perceptual hashing algorithms such as `pHash`, `PDQ`, and `PhotoDNA`, using LPIPS and L_2 as visual loss constraints can achieve a 100% attack success rate. Notably, using LPIPS as the visual loss typically allows the attack to be completed in less training rounds. For the deep learning-based perceptual hashing algorithm `NeuralHash`, using L_2 as the visual loss can increase the success rate of the attack.

Results. Table 3 presents the results of our triggering regulation attack against perceptual hashing algorithms. The Success Rate indicates the proportion of images that successfully collide in the hash space. Except for `NeuralHash`, all attacks achieved a 100% success rate. We also provide the visual distance between the original image X and the corresponding attacked image X', calculated using both L_2 and LPIPS. L_2 quantifies pixel-wise image changes, while LPIPS is more aligned with human visual perception of differences. Rounds represents the average number of optimization steps performed before a hash collision occurs. For the `PDQ` algorithm, on average, only 65 training rounds are required to generate a hash collision below the baseline $\Delta_d = 92$, significantly less than the 7k training rounds required by [38]. All perceptual hashing algorithms successfully complete the

Method	PH	Success Rate	Time↓	Rounds↓	L_2 Distance ↓	Δ_d	Hardware
Prokos	PhotoDNA	90%	4.00h	20000	42.81	1800	5888-Core 1.5-GHz GPU
ATKSCOPES	PhotoDNA	100%	0.50h	1518	95.57	1800	6-Core 3.0-GHz CPU
Prokos	PDQ	100%	0.10h	600-6000	77.62	90	36-Core 2.1-GHz CPUs
ATKSCOPES	PDQ	100%	0.07h	65	76.60	90	12-Core 3.6-GHz CPU

Table 4: Comparison to prior work on PhotoDNA and PDQ.

PH	γ	c	Success Rate	LPIPS Distance↓	L_2 Distance ↓	Time ↓	Δ_d
PDQ	1	10	100%	0.42	86.2	0.07h	92
PDQ	1	100	100%	0.43	86.5	0.06h	92
PDQ	0.1	10	100%	0.29	79.2	0.75h	92
PDQ	0.1	100	100%	0.30	79.2	0.71h	92

Table 5: Triggering regulation attack against PDQ with different γ and c values

attack within 1k training rounds. The attack progression is shown in Figure 8. Additionally, we show the attacked images generated using LPIPS and L_2 as visual loss constraints, as depicted in Figure 7.

5.4 Comparison to Prior Work

To prevent the source code from being exploited by malicious actors to compromise the legitimate rights of child protection agencies and related software users, previous research [17,38] closely related to ours chose not to disclose their source code. Therefore, we decide to compare with [38], which uses a more detailed description of attack effects.

We conduct the triggering regulation attack on PhotoDNA and PDQ, setting the thresholds at 1800 and 92, respectively, consistent with the experimental setup of Prokos et al. [38]. As shown in Table 4, our ATKSCOPES achieves less attacking time and training rounds, with less powerful hardware, implying clear efficiency gains. Attackers can use the most common and low-cost equipment, i.e. the single CPU of a personal computer, to carry out effective attacks. More importantly, the time cost of attacking each image is low enough – as for [38], 4 GPU hours for just one image, it is too long under realistic considerations. In addition, for attacks on PhotoDNA, our method achieves a 10% higher success rate compared to the approach proposed in [38], reaching a 100% attack success rate.

We should highlight that efficiency is not an irrelevant factor, it is closely related to the security of E2EE-PHM. In reality, the cost of an attack is always finite – a more efficient attack means that the training converges more easily, the probability of a successful attack is higher at fewer training rounds, larger scale attacks are possible, and cheap hardware can enable attacks as well.

5.5 Trade-offs Between Visual Quality and Attack Effectiveness

In this paper, we pay more attention to attack effectiveness (i.e., success rates and efficiency), as our purpose is to reveal vulnerabilities and their practical feasibility. We must acknowledge that there is an inevitable trade-off between visual quality and attack effectiveness in adversarial attacks. Improving visual quality typically means reducing the intensity of the adversarial perturbations, which may affect the success rate or efficiency of the attack. On the other hand, pursuing higher attack success rate or efficiency often requires stronger perturbations, which can lead to a decrease in visual quality.

In this regard, we offer two adjustable parameters, learning rate γ and loss function weight factor c. As shown in Table 5, we perform triggering regulation attacks against PDQ with different γ and c values. Here, γ is related to the step size for each update of the perturbation, where smaller γ implying a tendency to find the solution in a small neighborhood (and therefore better visual quality); c is related to the ratio of the visual loss to the total loss, where smaller c implying a tendency to visual quality in finding the solution.

6 Possible Countermeasures and Recommendations for Social Good

From our technical insights, we discuss the following possible countermeasures and recommendations for social good - they can be used together.

- **Fundamentally improving the representation robustness.** The most straightforward idea is to force the hashing to rely on more robust features. First, *symmetry priors* are introduced to enable a principled design of representations that achieve invariance to given forms of

perturbations. For example, representation redesign with geometric deep learning [6] or functional continuity [52] for invariance of basic geometric or signal transformations. Then, heuristic *data preprocessing* can force representations to describe more robust image features. For example, data preprocessing with simple smoothing filtering [30] or learned restoration network [56] partially destroys the perturbation pattern before extracting features. Finally, *adversarial training* [13] of the representation network can empirically enhance its robustness. For example, introducing training samples generated by ATKSCOPES for `NeuralHash` is a quick remediation [47].

- **Introducing more complexity into the system.** We note that efficient attacks are largely based on the simplicity/transparency of E2EE-PHM systems. Increasing the system complexity can enlarge the solution space for an attack to a computationally impractical level. A straightforward idea is to use *more complex representations*, such as larger feature sets, hierarchical representation designs, ensemble of heterogeneous representations, and sophisticated mappings from feature to hash. Another idea seeks *confidentiality or randomization mechanisms* with computational hardness assumption as a stronger remediation of adversarial vulnerabilities. This cryptographic strategy is often used in cyberspace security [46], for example, we can randomly select features in a large feature set for matching – the attack difficulty will depend on the size of the large feature set and the selected set.

- **Developing perturbation detectors as data preprocessing.** We can also develop perturbation detectors to determine whether the user-uploaded image has been perturbed. Since they work in a preprocessing form, such detectors can be easily combined with existing E2EE-PHM systems as a quick remediation of adversarial vulnerabilities. This idea has been proven effective in forensic research [58], for example, we can train a forensic network to learn discriminative properties between the natural and adversarial distributions, over a large set of natural images and their perturbed versions.

7 Expansion to Other Media Types

In this paper, we only discuss images as a typical media for E2EE-PHM. As our future work, the algorithm of ATKSCOPES can be extended almost directly to video scenarios, and the core idea of multiresolution perturbation has potential in other media such as audio.

Regarding the video, hashing methods generally extract features from video frames, followed by a fusion for a global hash [9, 51]. Moreover, existing adversarial attacks of video hashing act also on the pixel scale of video frames [4, 18]. Therefore the main motivation and implementation algorithm of this paper holds for video hashing – adding multiresolution perturbations to video frames is expected to exhibit similar attack gains.

Regarding other media such as audio, our algorithm does not work directly, but core idea of multiresolution perturbation is still useful as the multiscale structure of the data is almost universal [49]. For audio – although adversarial attacks on audio hashing have yet to appear [36] – we consider a one-dimensional version of Definition 1, by introducing a family of one-dimensional orthogonal functions. Then the main motivation and implementation algorithm of this paper will hold for audio, with possible effective attacks.

8 Conclusion

In this paper, we have systematically revealed the adversarial vulnerability of worldwide E2EE-PHM platforms, with potentially serious social implications. Unlike previous attacks at top-tier security conferences, our attack is a more realistic threat – *uniformly fooling* the famous `pHash`, Facebook `PDQ`, Microsoft `PhotoDNA`, and Apple `NeuralHash`, even with *higher success rates* and *less training times*, under both scenarios of *escaping* and *triggering* regulation (Formulations 1 and 2).

- At the **technical** level, we have proposed ATKSCOPES of *multiresolution perturbation*. With its formalized design (Definition 1), each perturbation element can affect image regions of adjustable scales – from the pixel scale (Example 2), to the mid scale (Example 3), and to the global scale (Example 1). Here, ATKSCOPES encapsulate previous pixel-scale-only attacks [14, 17, 29, 38] as special cases. Moreover, ATKSCOPES exhibit a huge leap in efficiency (Table 4 and Figure 8), compared to previous attacks with mismatched scales by Prokos et al. [38] (Section 5.4).

- At the **practical** level, we have achieved uniform, fast, and successful adversarial attacks as realistic threats to 4 commercial hashing algorithms in 2 scenarios (Section 5.3). From our technical insights, we have also discussed possible countermeasures and recommendations, i.e., fundamentally improving the representation robustness, introducing more complexity into the system, and developing perturbation detectors as data preprocessing (Section 6).

9 Ethical Considerations

Perceptual hashing is a crucial tool for regulating image content that violates basic human rights. Our study reveals potential vulnerabilities of such hashing techniques, exploring the

feasibility for escaping or triggering regulation. Intuitively, there is an ethical risk that our attack methods could be misused by malicious actors. On the other hand, the security community has long argued that openness and adversarial scrutiny are essential for ensuring the correct operation of systems, maintaining that "security by obscurity" is a fragile and ineffective approach to protecting high-value systems. This is because, without disclosing these vulnerabilities, the industry is often reluctant to invest in improvements. Therefore, the benefits of conducting this analysis to enhance security and resilience outweigh the potential risks associated with disclosing our findings.

To minimize potential ethical risks, we have implemented the following measures.

- **Technical countermeasures.** From our findings, we offer possible technical countermeasures: 1) fundamentally improving the representation robustness, 2) introducing more complexity into the system, and 3) developing perturbation detectors as data preprocessing. We will disclose our vulnerability findings, technical countermeasures, and source code to the relevant stakeholders, e.g., Apple, Facebook, and Microsoft, with the hope of collaborating on more robust E2EE-PHM systems.
- **Remediation period.** We set a remediation period for responsible disclosure practices. During this period, we will contact the relevant stakeholders with all necessary materials to repair the E2EE-PHM vulnerabilities, before an adversary gains the attack capability. During this period, our paper itself contains no sufficient implementation details, and the source code is only available to identified reviewers. After this period, we are willing to disclose the source code completely so that other researchers can validate and build upon it.
- **Wellbeing for team members.** Although our research focuses on the regulation of image content that violates basic human rights, we did not use any illegal images, such as child pornography, terrorism, drugs, and violent crime, during the study. This is because harmless images are equivalent to illegal images in terms of their E2EE-PHM processing. Therefore, our study does not pose any psychological harm to team members.

10 Compliance with the Open Science Policy

We have established a remediation period for responsible disclosure practices. After this period, we will make our source code publicly available on Zenodo https://zenodo.org/records/14633971, enabling other researchers to reproduce our work. To facilitate the reproduction process, the following artifacts will be provided:

- The dataset used in the experiments of this paper.
- The pHash, PDQ, PhotoDNA, and NeuralHash models.
- The LPIPS model for measuring perceptual image similarity.
- Executable Python files for escaping and triggering regulation attacks on pHash, PDQ, PhotoDNA, and NeuralHash.
- Readme files with instructions to run the attacks.

Acknowledgment

This work is partially supported by the Natural Science Foundation of China under Grant U24A20220, the Double Thousand Plan of Jiangxi Province under Grant jxsq2023201118, the Outstanding Youth Fund Program of Jiangxi Province under Grant 20232ACB212004, and the Guangdong Basic and Applied Basic Research Foundation under Grant 2024A1515012299.

References

[1] Mohammed Alkhowaiter, Khalid Almubarak, and Cliff Zou. Evaluating perceptual hashing algorithms in detecting image manipulation over social media platforms. In *2022 IEEE International Conference on Cyber Security and Resilience (CSR)*, pages 149–156, 2022.

[2] Apple. Apple csam detection. https://www.apple.com/child-safety/pdf/CSAM_Detection_Technical_Summary.pdf, 2021. Accessed on June 14, 2024.

[3] Apple. Expanded protections for children. https://www.apple.com/child-safety/, 2021. Accessed on June 20, 2024.

[4] Jiawang Bai, Bin Chen, Yiming Li, Dongxian Wu, Weiwei Guo, Shu-tao Xia, and En-hui Yang. Targeted attack for deep hashing based retrieval. In *Computer Vision–ECCV*, pages 618–634, 2020.

[5] Abhishek Bhowmick, Dan Boneh, Steve Myers, Kunal Talwar, and Karl Tarbe. The apple psi system. https://www.apple.com/child-safety/pdf/Apple_PSI_System_Security_Protocol_and_Analysis.pdf, 2024. Accessed on June 4, 2024.

[6] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[7] Johannes Buchner. Imagehash. https://github.com/JohannesBuchner/imagehash, 2021.

[8] Brian Dolhansky and Cristian Canton Ferrer. Adversarial collision attacks on image hashing functions. *arXiv preprint arXiv:2011.09473*, 2020.

[9] Zhen Dong, Su Jia, Tianfu Wu, and Mingtao Pei. Face video retrieval via deep learning of binary hash representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[10] Andrea Drmic, Marin Silic, Goran Delac, Klemo Vladimir, and Adrian S Kurdija. Evaluating robustness of perceptual image hashing algorithms. In *2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 995–1000, 2017.

[11] Facebook. Threatexchange github repository. https://github.com/facebook/ThreatExchange/tree/master/pdq. Accessed on June 14, 2024.

[12] Hany Farid. Reining in online abuses. *Technology & Innovation*, 19(3):593–599, 2018.

[13] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015.

[14] Qingying Hao, Licheng Luo, Steve TK Jan, and Gang Wang. It's not what it looks like: Manipulating perceptual hashing based applications. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 69–85, 2021.

[15] Shengshan Hu, Yechao Zhang, Xiaogeng Liu, Leo Yu Zhang, Minghui Li, and Hai Jin. Advhash: Set-to-set targeted attack on deep hashing with one single adversarial patch. In *Proceedings of the 29th ACM international conference on multimedia*, pages 2335–2343, 2021.

[16] INRIA. Inria copydays dataset. https://lear.inrialpes.fr/~jegou/data.php#copydays.

[17] Shubham Jain, Ana-Maria Cretu, and Yves-Alexandre de Montjoye. Adversarial detection avoidance attacks: Evaluating the robustness of perceptual hashing-based client-side scanning. In *USENIX Security Symposium*, pages 2317–2334, 2022.

[18] Linxi Jiang, Xingjun Ma, Shaoxiang Chen, James Bailey, and Yu-Gang Jiang. Black-box adversarial attacks on video recognition models. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 864–872, 2019.

[19] Jan Kaiser. jphotodna github repository. https://github.com/jankais3r/jPhotoDNA, 2021.

[20] Jan Kaiser. Pyphotodna github repository. https://github.com/jankais3r/pyPhotoDNA, 2021.

[21] Amin Kharraz, William Robertson, and Engin Kirda. Surveylance: Automatically detecting online survey scams. In *IEEE Symposium on Security and Privacy*, pages 70–86, 2018.

[22] Panagiotis Kintis, Najmeh Miramirkhani, Charles Lever, Yizheng Chen, Rosa Romero-Gómez, Nikolaos Pitropakis, Nick Nikiforakis, and Manos Antonakakis. Hiding in plain sight: A longitudinal study of combosquatting abuse. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 569–586, 2017.

[23] Evan Klinger and David Starkweather. phash: The open source perceptual hash library. https://www.phash.org/docs/, 2021. Accessed on June 24, 2024.

[24] Neal Krawetz. Photodna and its limitations. https://www.hackerfactor.com/blog/index.php?/archives/931-PhotoDNA-and-Limitations.html, 2021.

[25] Neal Krawetz. Tweet dated august 19, 2021. https://twitter.com/hackerfactor/status/1428395744705142785?s=20, 2021.

[26] Anunay Kulshrestha and Jonathan Mayer. Identifying harmful media in end-to-end encrypted communication: Efficient private membership computation. In *USENIX Security Symposium*, pages 893–910, 2021.

[27] Jie Li, Rongrong Ji, Hong Liu, Xiaopeng Hong, Yue Gao, and Qi Tian. Universal perturbation attack against image retrieval. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4899–4908, 2019.

[28] Xiaodan Li, Jinfeng Li, Yuefeng Chen, Shaokai Ye, Yuan He, Shuhui Wang, Hang Su, and Hui Xue. Qair: Practical query-efficient black-box attacks for image retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3330–3339, 2021.

[29] Xiaodan Li, Jinfeng Li, Yuefeng Chen, Shaokai Ye, Yuan He, Shuhui Wang, Hang Su, and Hui Xue. Qair: Practical query-efficient black-box attacks for image retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3330–3339, 2021.

[30] Bin Liang, Hongcheng Li, Miaoqiang Su, Xirong Li, Wenchang Shi, and Xiaofeng Wang. Detecting adversarial image examples in deep neural networks with adaptive noise reduction. *IEEE Transactions on Dependable and Secure Computing*, 18(1):72–85, 2018.

[31] Jordan Madden, Moxanki Bhavsar, Lhamo Dorje, and Xiaohua Li. Assessing the adversarial security of perceptual hashing algorithms. *arXiv preprint arXiv:2406.00918*, 2024.

[32] Microsoft. Photodna. https://www.microsoft.com/en-us/photodna, 2018. Accessed on June 24, 2024.

[33] Microsoft. Photodna cloud service terms of use. https://www.microsoft.com/en-us/PhotoDNA/TermsofUse, 2021. Accessed on June 14, 2024.

[34] NCMEC. NCMEC's Statement Regarding End-to-End Encryption. https://www.missingkids.org/blog/2019/post-update/end-to-end-encryption, March 2019. Accessed on June 14, 2024.

[35] U.S. Department of Justice. Attorney general barr signs letter to facebook from us, uk, and australian leaders regarding use of end-to-end encryption. https://www.justice.gov/opa/pr, 2019. Accessed on June 14, 2024.

[36] Hamza Özer, Bülent Sankur, Nasir Memon, and Emin Anarım. Perceptual audio hashing functions. *EURASIP Journal on Advances in Signal Processing*, 2005:1–14, 2005.

[37] Sergio Pastrana, Alice Hutchings, Daniel Thomas, and Juan Tapiador. Measuring ewhoring. In *Proceedings of the Internet Measurement Conference*, pages 463–477, 2019.

[38] Jonathan Prokos, Neil Fendley, Matthew Green, Roei Schuster, Eran Tromer, Tushar Jois, and Yinzhi Cao. Squint hard enough: attacking perceptual hashing with adversarial machine learning. In *USENIX Security Symposium*, pages 211–228, 2023.

[39] Shuren Qi, Yushu Zhang, Chao Wang, Jiantao Zhou, and Xiaochun Cao. A principled design of image representation: Towards forensic tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5):5337–5354, 2022.

[40] M Zubair Rafique, Tom Van Goethem, Wouter Joosen, Christophe Huygens, and Nick Nikiforakis. It's free for a reason: Exploring the ecosystem of free live streaming services. In *Network and Distributed System Security Symposium*, 2016.

[41] O. Russakovsky, J. Deng, H. Su, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[42] Kavita Saini, Vaibhav Agarwal, Arjun Varshney, and Anushka Gupta. E2ee for data security for hybrid cloud services: A novel approach. In *2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*, pages 340–347, 2018.

[43] SCSC under the MOND. Assessment of cybersecurity of mobile devices supporting 5g technology: Analysis of products made by huawei, xiaomi and oneplus. https://www.nksc.lt/doc/en/analysis/2021-08-23_5G-CN-analysis_env3.pdf, 2021. Accessed on June 14, 2024.

[44] Priyanka Singh and H. Farid. Robust homomorphic image hashing. In *CVPR Workshops*, pages 11–18, 2019.

[45] Lukas Struppek, Dominik Hintersdorf, Daniel Neider, and Kristian Kersting. Learning to break deep perceptual hashing: The use case neuralhash. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, pages 58–69, 2022.

[46] Chao Wang, Shuren Qi, Zhiqiu Huang, Yushu Zhang, Rushi Lan, Xiaochun Cao, and Feng-Lei Fan. Spatial-frequency discriminability for revealing adversarial perturbations. *IEEE Transactions on Circuits and Systems for Video Technology*, 2024.

[47] Xunguang Wang, Zheng Zhang, Guangming Lu, and Yong Xu. Targeted attack and defense for deep hashing. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2298–2302, 2021.

[48] Xunguang Wang, Zheng Zhang, Baoyuan Wu, Fumin Shen, and Guangming Lu. Prototype-supervised adversarial network for targeted attack of deep hashing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16357–16366, 2021.

[49] E Weinan and Bjorn Engquist. Multiscale modeling and computation. *Notices of the AMS*, 50(9):1062–1070, 2003.

[50] WhatsApp. Connecting one billion users every day. https://blog.whatsapp.com/connectingone-billion-users-every-day, 2017. Accessed on June 18, 2024.

[51] Gengshen Wu, Jungong Han, Yuchen Guo, Li Liu, Guiguang Ding, Qiang Ni, and Ling Shao. Unsupervised deep video hashing via balanced code for large-scale video retrieval. *IEEE Transactions on Image Processing*, 28(4):1993–2007, 2019.

[52] Chong Xiang, Arjun Nitin Bhagoji, Vikash Sehwag, and Prateek Mittal. Patchguard: A provably robust defense against adversarial patches via small receptive fields and masking. In *USENIX Security Symposium*, pages 2237–2254, 2021.

[53] Yanru Xiao and Cong Wang. You see what i want you to see: Exploring targeted black-box transferability attack for hash-based image retrieval systems. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1934–1943, 2021.

[54] Erkun Yang, Tongliang Liu, Cheng Deng, and Dacheng Tao. Adversarial examples for hamming space search. *IEEE transactions on cybernetics*, 50(4):1473–1484, 2018.

[55] Christoph Zauner. Implementation and benchmarking of perceptual image hash functions. Master's thesis, 2010. Available at: https://www.phash.org/docs/pubs/thesis_zauner.pdf.

[56] Kai Zhang, Yawei Li, Wangmeng Zuo, Lei Zhang, Luc Van Gool, and Radu Timofte. Plug-and-play image restoration with deep denoiser prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):6360–6376, 2021.

[57] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[58] Shuhai Zhang, Feng Liu, Jiahao Yang, Yifan Yang, Changsheng Li, Bo Han, and Mingkui Tan. Detecting adversarial data by probing multiple perturbations using expected perturbation score. In *Proceedings of the 40th International Conference on Machine Learning*, pages 41429–41451, 2023.

Appendix

In this section, we will introduce the working schemes of the four perceptual hashing algorithms used in this paper.

pHash first resizes the image to 32×32 and applies a grayscale transformation to convert it to a grayscale image. Then it applies the Discrete Cosine Transform (DCT) to the image and uses the first 8×8=64 low-frequency DCT coefficients to compute the DCT hash. Finally, it calculates the median of the DCT coefficients and generates a binary hash string based on whether each DCT frequency is above or below the median.

PhotoDNA [32] was invented by Microsoft Research and Dartmouth College in 2009, with the goal of combating child exploitation in collaboration with the National Center for Missing and Exploited Children (NCMEC). While the detailed implementation of `PhotoDNA` has not been officially published, a more comprehensive algorithm description can be pieced together from partial descriptions by Microsoft and information obtained through reverse engineering [24, 25].

In 2020, researchers [19, 20] extracted the binary implementation of `PhotoDNA` from forensic software . According to Krawetz [24], `PhotoDNA` first resizes the image to 26×26 pixels and converts it to grayscale. These pixels are then divided into overlapping 6×6 "grids", allowing the algorithm to produce nearly identical hashes even if the image is cropped by approximately 2%. Each "grid" is processed using the Sobel operator, which generates its hash code. The hash code of each grid is a 4-tuple representing the sums of the Sobel outputs to the left, right, up, and down. The final hash consists of 36 "grids" of 4-tuples, with a total length of 144. Unlike other Perceptual hash algorithms, the 144 hash codes output by `PhotoDNA` are integers ranging from 0 to 255.

PDQ [11]was introduced by Facebook in 2019 with the aim of developing photo and video matching technologies to enhance internet security.

`PDQ` first normalizes the brightness data of input images and then downsamples the images to 64×64 pixels using two Jarosz filters. Subsequently, it applies Discrete Cosine Transform (DCT) to the preprocessed images and selects the low-frequency DCT coefficients of the top 16×16 block, which capture the main features of the images. For each 16×16 block in the output hash, `PDQ` outputs 1 if the corresponding DCT coefficient is greater than the median, otherwise it outputs 0. The final output digest is a 256-bit binary string.

NeuralHash [2] was proposed by Apple in 2021 to accurately detect and report known child sexual abuse material (CSAM) in iCloud Photos accounts of their iCloud users. The system generates `NeuralHash` in two steps. First, an image is input into a convolutional neural network (CNN) which extracts visual features and encodes them into an N-dimensional feature vector, an abstract numerical interpretation of the image's characteristic features. Next, locality-sensitive hashing (LSH) is used to assign close feature vectors to buckets with similar hash codes. The feature vectors are transformed by computing the product of the hashing matrix and the embedding. The real-valued vector is then converted to a bit vector using the Heaviside step function, producing a 96-bit binary hash vector.