STRENGTHENING AUTHENTICITY AND MITIGATING MISINFORMATION

SLIGHTLY HOMOMORPHIC DIGITAL SIGNATURES AND PRIVACY PRESERVING FOLDING SCHEMES

SIMON ERFURTH

DECEMBER 2024

SDU
DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE

DDC
digital democracy centre

Strengthening Authenticity and Mitigating Misinformation

Slightly Homomorphic Digital Signatures and Privacy Preserving Folding Schemes

A dissertation
Presented to the Faculty of Science at University of Southern Denmark
In partial fulfillment of the requirements for the PhD degree

Simon Erfurth

Advisors: Joan Boyar
Kim S. Larsen
Ruben Niederhagen
Claes de Vreese

December 2024
Version 1.0

ABSTRACT

In an era where misinformation proliferates rapidly, ensuring the authenticity of digital content can be paramount. This dissertation explores two cryptographic solutions, which can be used to strengthen authenticity and thereby mitigate misinformation: *slightly homomorphic digital signatures* and *privacy preserving folding schemes*.

The first part of this dissertation focuses on slightly homomorphic digital signatures. Specifically, we construct quotable signatures for text, and digital signatures for images allowing JPEG compression. Quotable signatures allow extraction of a signature for a quote from a text, from a signature for the text, ensuring that quotes can be authenticated, even when detached from their original context. The digital signature scheme for JPEG images supporting compression is similar. It allows extracting a signature for a compressed JPEG image, from a signature for the original image, despite the utilized JPEG compression being lossy. Our construction requires the used quantization tables to contain only powers of two. For both constructions, the extracted signature is signed with the same private key as the original signature and, crucially, extraction does not require knowledge of the key, nor interaction with the signer.

In the second part, we introduce privacy preserving folding schemes, a natural extension of folding schemes with selective verification. Folding schemes transform the task of creating multiple zero-knowledge proofs that statements are in a language into creating one zero-knowledge proof for a (new) statement from the same language, at the cost of verification also requiring one to check a (cheap) inclusion proof. With known constructions of folding schemes, the inclusion proofs for a statement leak other statements. Privacy preserving folding schemes ensures that verification of one statement does not leak information about other statements, at a minimal increase in inclusion proof size. This is achieved through the introduction of *NP-statement hiders*, which allow an instance of a relation to be hidden as a new instance in the same relation, in a verifiable way.

We define and prove the security and the efficiency of these cryptographic constructions through rigorous theoretical analysis and performance evaluation. The proposed constructions offer mechanisms for maintaining the integrity and authenticity of digital content, providing a step forward in the fight against misinformation.

RESUMÉ

I en tid, hvor misinformation florerer som aldrig før, er det afgørende at sikre ægtheden af digitalt indhold. Denne afhandling undersøger to kryptografiske løsninger, der kan bruges til at styrke ægthed og modvirke misinformation: *slightly homomorphic digital signatures* og *privacy preserving folding schemes*.

Den første del af afhandlingen fokuserer på slightly homomorphic digital signatures. Specifikt konstruerer vi citerbare signaturer for tekst- og billedesignaturer, der tillader JPEG-komprimering. Med citerbare signaturer kan man udlede en signatur for et citat fra en tekst, fra en signatur for teksten, således at citater kan verificeres uden den tekst, de oprindeligt stammer fra. Vores billedesignatur, der understøtter JPEG-komprimering, gør, at man kan udlede en signatur for et komprimeret JPEG-billede fra en signatur for det originale billede. Dette på trods af at den anvendte JPEG-komprimering er destruktiv. Vores konstruktion kræver, at de anvendte kvantiseringstabeller kun indeholder toerpotenser. For begge konstruktioner er den udledte signatur underskrevet med den samme private nøgle som den oprindelige signatur, og udledningen kræver ikke kendskab til den private nøgle eller interaktion med underskriveren.

I den anden del af afhandlingen introduceres privacy preserving folding schemes, som er en naturlig tilføjelse til folding schemes med selektiv verifikation. Med et folding scheme kan man i stedet for at generere flere zero-knowledge-beviser for, at en række udsagn er i et sprog, generere ét zero-knowledge-bevis for, at et udsagn er i det samme sprog. Dette dog med den omkostning at verifikation så også kræver, at man tjekker et inklusionsbevis. Når det, at udsagnene er i sproget, skal bevises over for forskellige verifikatorer, kan inklusionsbeviserne lække andre verifikatorers udsagn. Privacy preserving folding schemes sikrer med en minimal forøgelse af inklusionsbevisets størrelse, at verifikation af et udsagn ikke lækker information om andre udsagn. Dette opnås gennem introduktionen af *NP-statement hiders*, med hvilke en instans af en relation kan skjules som en ny instans af den samme relation på en verificerbar måde.

Vi definerer og beviser sikkerheden og effektiviteten af disse kryptografiske konstruktioner gennem grundig teoretisk analyse og ydelsesevaluering. De foreslåede konstruktioner kan bruges som værktøjer til at opretholde integriteten og ægtheden af digitalt indhold og kan være et skridt fremad i kampen mod misinformation.

PUBLICATIONS

This thesis is based on the following publications, which are presented as Chapters 2 to 4.

[BELN23] Joan Boyar, Simon Erfurth, Kim S. Larsen, and Ruben Niederhagen. Quotable signatures for authenticating shared quotes. In *Progress in Cryptology - LATINCRYPT 2023*, volume 14168 of *Lecture Notes in Computer Science*, pages 273–292. Springer, 2023. DOI: 10.1007/978-3-031-44469-2_14.

[Erf24] Simon Erfurth. Digital signatures for authenticating compressed JPEG images. In *Security-Centric Strategies for Combating Information Disorder - SCID 2024*, page 4. ACM, 2024. DOI: 10.1145/3660512.3665522.

[BE24] Joan Boyar and Simon Erfurth. Folding schemes with privacy preserving selective verification. *IACR Communications in Cryptology*, 1(4), 2024. URL: https://eprint.iacr.org/2024/1530.

During my PhD I have also been part of the following interdisciplinary works, which can be found in Appendices A and B.

[GE23] Marília Gehrke and Simon Erfurth. Adding quotable signatures to the transparency repertoire in data journalism. In *Joint Computation+Journalism Symposium and European Data & Computational Journalism Conference 2023*, 2023. URL: https://www.datajconf.com/papers/CJ_DataJConf_2023_paper_17.pdf.

[EEG25] Johanna Eggers, Simon Erfurth, and Marília Gehrke. Image authenticity in the age of AI: digital signatures as a defense against visual disinformation, 2025. Submitted to Cambrdige Disinformation Summit 2025.

For articles [BELN23; BE24; GE23; EEG25], I state my proportionate contribution in Appendix C, in accordance with the PhD school's guidelines.

ACKNOWLEDGMENTS

First, I would like to thank my supervisors: Joan Boyar, Kim S. Larsen, Ruben Niederhagen, and Claes de Vreese. Both for taking me on as a PhD student (and Joan for reaching out to me, asking if I was interested in doing a PhD in the first place), and in particular for teaching me much more than "just" how to conduct research in computer science.

A huge thanks also go out to the entire Department of Mathematics and Computer Science (IMADA), where I have been both a student and an employee for the last eight years. When I finished my master's thesis, I remember feeling happy that I was done with the thesis, but also sad that my time at IMADA was at an end. Luckily, I got the chance to return, and I have been just as happy to be at IMADA over the last three years, as I was the first five. Special thanks to those I have shared an office with over the years, first the guys and girls at the IMADA TA office Balkonen (you know who you are!), and for the last almost three years, Magnus Berg, with whom I have had many insightful discussions.

I would also like to thank all the members of the Digital Democracy Centre at SDU, who have shared many good times with me, both in academic settings, such as the PhD/Postdoc club, and in social settings out in the city or at one of our great retreats. In particular, I am grateful the members of the Trust and News Authenticity project: Johanna Eggers, Marília Gehrke, David N. Hopmann, and my supervisors. Without you, my PhD project would not have been as interdisciplinary nor as well-founded in real world issues.

Outside the world of academia, a big thanks to current and former members of Odense University Rowing, who at times have been my lifeline to a world outside digital signatures, hash functions, and questions about authenticity. Similarly, I would like to thank the people in "the meme chat" (you definitely know who you are!) for the many jokes, and for always being ready with a classy deez or ligma joke.

Last, but definitely not least, I would like to thank my family and my girlfriend for always being there for me. Your unwavering support throughout the PhD process has meant more to me than words can express. There are so many things I am grateful for that I can't even begin to list them all.

CONTENTS

1 INTRODUCTION

A natural way to get an idea about both the societal issues motivating undertaking this PhD project, and the technical contributions made as part of this PhD project, is to dissect the title of my thesis:

"Strengthening Authenticity and Mitigating Misinformation – Slightly Homomorphic Digital Signatures and Privacy Preserving Folding Schemes"

The first part – *Strengthening Authenticity and Mitigating Misinformation* – relates to the motivation for carrying out my PhD project in the first place. While mis- and disinformation cannot be considered new phenomena by any means, the last decade has seen a change to how news is consumed, where it is more and more consumed via a social network, rather than directly from various news media (in a survey of over 95,000 people in 47 countries, only 22% of people have direct access to news media as their main gateway to online news [NFR+24]). Parallel with this change, it has become increasingly hard to discern the fake from the real, and over half of people are now concerned about what is real and what is fake when it comes to news online [NFR+24]. The Digital Democracy Centre at SDU initiated the *Trust and News Authenticity* project to investigate an alternative to the traditional "fact checking"-approach. Rather, the project idea was to instead mark content originating from quality news media as such, thereby strengthening the authenticity of quality content and, hopefully, contribute to mitigating the effects of misinformation online. This lead to two project tracks, where one track aims to investigate how to visually label quality content as such and the effects of such labeling, and another track works to find technical solutions for how such labeling could be done. My PhD has been part of the second track, which leads to the next part of the thesis title.

Our initial idea for labeling content shared on social media was to use *Slightly Homomorphic Digital Signatures*. For our use case, slightly homomorphic signatures, or P-homomorphic signatures [ABC+15], can be thought of as digital signatures with the additional property that if $P(m, m') = 1$ for some predicate P, then *anyone* can derive a signature for m' from a signature for m.[1] Crucially, the signature for m' is signed with the same key as the signature for m, despite the derivation procedure not requiring knowledge of this key. As part of this project, we have worked with two different concrete instantiations of slightly homomorphic signatures. The first slightly homomorphic signature scheme is *quotable signatures*, and the predicate is 1 if m' is contained as a quote in m. The second

1 In [ABC+15], they more generally let the predicate relate a set of messages M to a single message m', and additionally require that the derived signature for m' reveals no information about m that cannot already be derived from m'. Thus, our variants can be considered a form of weak slightly homomorphic signatures.

slightly homomorphic signature scheme is a signature scheme for JPEG images allowing JPEG compression. Here, the predicate is 1 if m' can be obtained by compressing m with parameters from a specific family. We focused on compression, since images are almost always compressed when they are uploaded to social media, which makes compression a very common image transformation. For both instantiations, we define explicit unforgeability notions, and constructe concrete schemes, which we prove are secure with respect to the relevant unforgeability notions, and which we analyze the efficiency of. Our quotable signature scheme builds on a folklore idea [ABC+15], involving creating a Merkle tree over the text, concretely suggested as a potential way to mitigate misinformation in [KNSS19]. The image signature scheme is inspired by the ideas in [JWL11], but suppors only compression, rather than a larger selection of transformations, resulting in a scheme that is significantly more usable in practice.

Finally, *Privacy Preserving Folding Schemes* is an extension of folding schemes with selective verification [RZ23], which is again an extension of folding schemes [KST22]. Suppose that a prover wishes to prove that for several statements $x_1,\dots,x_n$ it knows witnesses $w_1,\dots,w_n$, without revealing the witnesses. One way to do this would be to generate n zero-knowledge succinct non-interactive arguments of knowledge (zk-SNARKs), or some other flavor of zero-knowledge proofs. However, generating n zk-SNARKs is relatively costly. As an alternative, a folding scheme allows the prover to *fold* together the n statements/witness pairs into one pair (x,w), such that w is a witness for x, if and only if the prover knows witnesses for each of the n statements. The prover can then use a zk-SNARK to prove knowledge of w, and additionally prove that x was formed by folding together the claimed statements, using a folding proof π, which the folding scheme also generates. Originally, folding schemes were used as a part of incrementally verifiable computing [KST22], where the verifier wishes to verify the validity of all initial statements. For different applications, such as distributed computation, there may be many provers, each only interested in verifying the validity of one statement, and hence not required to verify each of the $n-1$ other statements used when forming x [RZ23]. For these applications, folding schemes with selective verification are more efficient, since they generate n selective proofs of folding, each only verifying that one specific statement x_i is included in x. One issue remains: a selective proof of folding might still leak information about statements, other than the one it corresponds to.[2] We define and construct privacy preserving folding schemes,[3] which extends folding schemes with selective verification by guaranteeing the privacy of all statements, other than the one being verified. Circling back to the societal motivation for the project, privacy

2 Specifically, in the original construction of folding schemes with selective verification [KST22], a selective proof of folding for x_i leaks either x_{i-1} or x_{i+1}.

3 In [BE24] we name these schemes *folding schemes with privacy preserving selective verification*, but since it does not make sense to consider folding schemes with privacy preserving but non-selective verification, we have since stopped specifying "selective verification".

preserving folding schemes have potential applications in verifying how images have been transformed, which, similar to our slightly homomorphic signatures for JPEG images, can be part of an approach to strengthening authentic content online [DEH25].

In this introductory chapter, we will return to the technical contributions of the project in Sections 1.1 and 1.2, and the second part of this thesis (from Chapter 2 and onward) contains the original manuscripts, describing the technical contributions in greater detail. For now, we will circle back to the societal motivation for the project, and attempt to both further motivate the contributions of this project, and place them into a broader context.

The Rise of Mis- and Disinformation

As we mentioned earlier, mis- and disinformation are not new phenomena. An early example of a successful disinformation campaign is the 44 B.C. smear campaign by Octavius against Marcus Antonius, which included writing statements on coins, painting Antonius as a drunk and a womanizer under the influence of Cleopatra [PM18]. Octavius of course became Augustus, the first Roman emperor. While this shows that disinformation has been possible for thousands of years, it also serves as an illustration of the effort once required to mass spread disinformation. Every coin would have to be engraved by hand, making disinformation campaign at a large scale a massive effort.

The invention of the printing press in 1498 greatly lowered the bar for creating and distributing mis- and disinformation, by enabling cheap and rapid duplication of texts. A classic example of this can be found in the "Great Moon Hoax" of 1835, wherein the New York newspaper "The Sun" published a series of articles, claiming that life had been found on the moon, including fabricated pictures of bat-like winged humans and unicorns, see Figure 1 [Tho00]. The development of one-to-many communication forms, such as radio and TV, only further lowered the bar for spreading mis- and disinformation. Here the classic misinformation example is the 1938 transmission of a radio drama adaptation of Herbert Wells' novel "The War of the Worlds", which some listeners mistook as a real thing, leading to numerous calls to police, newspaper offices, and radio stations [Sch15].

While the aforementioned developments all lowered the bar for creating and distributing mis- and disinformation, it is probably fair to say that their impact vanishes in comparison with the many-to-many communication enabled first by the introduction of the internet, and then by the development of social media [PM18]. Not only did this lower the bar for what was required to create and distribute (mis-/dis-) information, even before the rise of generative AI, but it also enabled friction-free creation of more or less ad-hoc groups of people with similar views, increasing the size and reach of echo chambers where mis- and disinformation agreeing with the groups' prevalent opinions can flow effortlessly, and

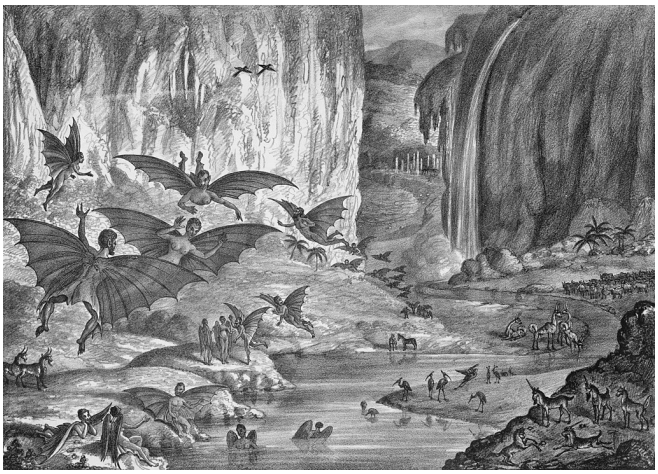


Figure 1: Illustration from the "Great Moon Hoax" series of articles published by The Sun (New York) in 1835 [Tho00].

where more nuanced content might be unable to gain meaningful traction [IP18; RN23]. As the final ingredient for a perfect storm, OpenAI introduced the world at large to generative AI, when they released ChatGPT in 2022 [Nys24], which has since been followed by numerous other image and text generating AIs. Generative AI allows almost anyone to create convincing looking misleading content in volumes that just a few years ago would have been almost unimaginable, outside a government funded propaganda project.

Given these conditions, it is no surprise that many are concerned about fake news, and what it might mean for them. The already mentioned Reuters Digital News Report surveyed over 95,000 people in 47 countries, representing half of the world's population, and found that 59% of people are concerned about what is real and what is fake when it comes to news online, up from 54% in 2022 and 55% in 2019 [NFR+24; NFR+22; NFKN19]. Part of the reason for this increasing concern might be attributed to news consumption increasingly moving away from formats directly belonging to various news media, such as physical newspapers and websites, and instead moving to third party distributors, such as social media and search engines, which is known to make it harder to recall from what news brand a given story originates [KFN18], and news brands' reputation serve as an important heuristic when evaluating the quality of a story [US14]. As mentioned, only 22% of people directly access news media as their main gateway to online news, and this proportion has been on a steady decline down from 32% in 2018 [NFR+24]. Instead, news is increasingly both found through and consumed on social media platforms, with most social media[4] having a steady increase in how large a proportion of people use them for news over the last 10 years [NFR+24].

4 With the main exception being Facebook, which have made algorithmic changes deprioritizing news [NFR+24].

Two very common and relatively concrete approaches to mitigating the consequences of mis- and disinformation are fact checking individual stories and reliability ratings of news outlets. Fact checking requires detecting an individual story or trend, checking it and, usually, labeling the story where it appears. Each step can either be done automatically or manually. Two different examples of this is the Community Notes system on X (formerly known as Twitter), where users can suggest community notes for a post (detecting), and if other users find them helpful (checking) they appear on that specific post (labeling), and Facebook's approach, where they work with third party fact checking organizations to identify misinformation trends (detecting and checking), and then, using automated systems, mark every post related to the trend as potentially misleading (labeling).[5] One issue with the fact checking approach is that, by its nature, it is always playing catch up, since trends have to be detected and checked before they can be marked as problematic [VRA18]. With reliability ratings, the approach is to instead rate if a news *outlet* is trustworthy or not, rather than focusing on a specific story or trend. One example of using reliability ratings to combat the consequences of fake news is *NewsGuard Ratings*.[6] NewsGuard has rated over 35,000 news sources. Reliability ratings avoid having to play catch up with each story, by having rated many outlets, and then applying an outlet's rating to all stories from the outlet. While this could lead to replacing trying to catch up with individual stories with trying to catch up with new outlets, NewsGuard Ratings attempts to avoid this by not just rating unreliable outlets, but also rating reliable outlets, so that the absence of a rating is itself a warning. One large issue with reliability ratings is that it requires content to be linked to a publisher in order to rate it. Another issue, that affects both the fact checking and reliability rating approach, is that it has been shown that just flagging potential "fake" content as such (*negative labeling*) tends to increase the negative effects of fake news, rather than mitigate them [DSÁ20; LES+12; SK20].

Cryptography Against Fake-News

This leads us to the fundamental idea behind the Trust and News Authenticity project. The project aims to explore an approach where individual pieces of news content are labeled with their source, and where cryptographic tools are used to guarantee the authenticity of the content, relative to the signing source. Thus, our approach can be thought of as applying *positive labeling* to authentic content from quality sources, rather than the fact checking approach which applies negative labeling, and the reliability rating which applies both positive and negative labeling, but only in selective cases (for NewsGuard Ratings, only when there is a direct link to the outlet, the outlet has been rated, and currently without

5 See https://communitynotes.x.com/guide/en/about/introduction and https://www.facebook.com/help/1952307158131536.

6 https://www.newsguardtech.com/solutions/news-reliability-ratings/

any solution to verifying the integrity of the content). One part of the project worked on the effects labeling had on user behavior and what users interpret labeling to mean, see for example [GEdVH24]. The part I have been most involved in, instead focused on creating technical solutions, allowing positive labels to follow content as it is transformed and shared, in particular on social media, while still verifying the authenticity of the content.

For a cryptographer, the project focus on the authenticity and integrity of content immediately lead one's thoughts to signature schemes. A trivial "solution" would be to have news outlets create signatures for their content, and then (somehow) share these signatures together with the content. While not the only challenge with the trivial approach, one fundamental issue is that standard digital signatures require the message being verified to be bit-for-bit identical to the signed message, and hence do not allow verifying just parts of a signed article, or compressed versions of a signed image. At the very core, this issue has been a common thread connecting my research throughout this project. Slightly homomorphic digital signatures is a special version of digital signatures, which can be constructed to allow exactly the sort of modifications that we wish to allow. Folding schemes, on the other hand, have already found applications in solutions for proving that an image has only been modified in specific ways [DEH25]. If one allows, for example, only color corrections and compression, this can be used as the authenticity checking part of our positive labeling solution.

Naturally, a number of different approaches for cryptographic tools to be used to mitigate the negative effects of mis- and disinformation has been considered over the years. Our work on slightly homomorphic signatures for text builds on ideas from [KNSS19], wherein they propose using digital signatures allowing quotations as a tool for mitigating fake news. Another work, focusing on adding digital signatures to multimedia content on social media in order to defend against fake news, and hence more related to our slightly homomorphic signatures allowing JPEG compression, is [AJAZ22]. Their work focuses more on the technical aspects of how to include digital signatures on social media platforms, and in particular only suggests using "standard" digital signatures (and hence disallowing compression). A different direction of research goes into which properties a cryptographic system should have, in order to most efficiently mitigate mis- and disinformation. As an example of this, [SIM+22] investigates *cryptographic provenance systems* for mitigating misinformation, drawing on both literature from journalism, human-computer interaction, and cryptography. They specifically use the term cryptographic provenance system to mean a system combining a cryptographic system with some further specified properties with a usable interface. A very different direction is taken in [WCM24], where they draw on methods from formal verification to propose a different system for determining and tracking the provenance of the news and social media.

One notable project, also advocating for the use of positive labeling in addressing misinformation, is the Coalition for Content Provenance and Authenticity (C2PA) project.[7] This project involves many companies, including Adobe, Amazon, BBC, Google, Meta, Microsoft, and OpenAI, to name some of the major ones. The C2PA project is a unification of the Adobe lead Content Authenticity Initiative and the joint Microsoft and BBC lead Project Origin. The goal of this project is to develop and promote adaptation of an open standard, which supports tracing the provenance of content, which the C2PA defines as "basic, trustworthy facts about the origins of a piece of digital content." Examples of provenance could be who created it and when (here it is worth mentioning that Leica recently released a camera, in collaboration with the C2PA, which is supports signing pictures with C2PA credentials, as they are captured [Lyo23]), how/by whom/when it was edited, and even the combined provenance of two merged prices of content. Roughly, the idea is then to have content signed by C2PA compatible programs and devices, when it is captured and later edited, and then having the credentials verified when the content is displayed, and even allowing the user to dive deeper into the credentials, if they so desire.

We now move on to presenting the technical contributions of this project, as well as placing them in the context of the relevant literature surrounding them. Section 1.1 contains our work on slightly homomorphic digital signatures for both text and images, corresponding to Chapters 2 and 3, and Section 1.2 gives an overview of privacy preserving folding schemes, corresponding to Chapter 4.

1.1 SLIGHTLY HOMOMORPHIC DIGITAL SIGNATURES FOR QUOTING TEXT AND COMPRESSING IMAGES (CHAPTERS 2 AND 3)

In general, a traditional signature scheme is triple of algorithms, (KeyGen, Sign, Verify). The canonical digital signature reference is the seminal 1976 work by Diffie and Hellman, "New Directions in Cryptography" [DH76], but recent standards can be found in [Nat24a; Nat24b]. The first algorithm is a key generation algorithm, often denoted KeyGen, which generates pairs of related keys $(\mathsf{pk}, \mathsf{sk})$, where pk is called the *public key* and sk is called the *secret key*. The second algorithm is a signing algorithm Sign, which, given a message m and a secret key sk, generates a *signature* σ. We say that σ is a signature for m under secret key sk. The third algorithm Verify is a verification algorithm, and takes as input a public key pk, a message m, and a signature σ. The verification algorithm verifies if σ is a valid signature for m under the secret key sk corresponding to pk.

Typically, signature schemes are required to (1) be complete, meaning that if a key pair $(\mathsf{pk}, \mathsf{sk})$ is generated by KeyGen, and a message m is signed using sk, then Verify will (with overwhelming probability) accept the signature as valid for m with respect to pk. (2) The scheme is also required to be unforgeable, essentially meaning that without knowing

7 https://c2pa.org

the secret key from a key pair, it is impossible to forge a signature for a message that Verify will accept under the public key, except with negligible probability. Note that for unforgeability, there are both different notions as to "how" unforgeable a scheme should be (for example universal forgery and existential forgery), and different attack models (for example key only attack and chosen message attack). It is common to require that a signature scheme is at least existentially unforgeable under chosen message attacks (EUF-CMA) [Nat24a; Nat24b].

For the Trust and News Authenticity project, the first idea we worked on was signature schemes for text, allowing quoting parts of the text, such that the signature could follow the quote as it was shared on social media or in news outlets, acting as an authenticity indicator. Already here there are two potential issues with traditional signature schemes. First, traditional digital signatures require the message being verified to be bit-for-bit identical to the message that was signed, meaning that when quoting text, one would still need to provide the remainder of the text that was signed, but not quoted. Similar issues would often apply to other forms of media. The second issue, is that in the common version of the EUF-CMA security definition for signature schemes, a signature scheme is insecure if an adversary can output a message it has not obtained from the environment (usually modeled using a signing oracle), and a valid signature for the message. If we create a signature scheme where a signature for a message is also valid for a quote from the message (or where a valid signature for the quote can be derived from the original signature), an adversary could just choose a message, obtain a signature for the message from the environment, quote part of the message, and output the quote and signature for the quote, thereby breaking EUF-CMA security. Again, this issue would also apply to other forms of media.

To remedy these issues, our approach uses signature schemes that allow some transformation of the content, and defines alternative security notions, that additionally require the message the adversary outputs to not be derivable (by quoting/compression) from any message queried to the oracle. Both our signature schemes fit into the slightly homomorphic signature framework, proposed by Ahn et al. [ABC+15] in 2011, which unifies several different concepts, such as quotable, redactable, arithmetic, and transitive signatures. Slightly a homomorphic signature scheme extends a traditional signature scheme as follows. Let $\mathcal{M}$ be the message space of the signature scheme, and

$$P\colon \mathcal{P}(\mathcal{M}) \times \mathcal{M} \to \{0,1\} \tag{1}$$

a predicate mapping a set of messages and a message to a bit. For a message $m' \in \mathcal{M}$, and a set $M \subseteq \mathcal{M}$, a slightly homomorphic signature scheme allows *anyone* to derive Alice's signature for m' from Alice's signatures for M, if $P(M, m') = 1$, in which case m' is also said to be derivable from M. For this reason, slightly homomorphic signatures are also called P-homomorphic signatures. Slightly homomorphic signatures' se-

curity definition includes two properties: unforgeability and context hiding. The unforgeability notion is a natural extension of the traditional notion, where the message the adversary outputs should neither be obtained from the environment, nor derivable from any subset of the set of messages it has obtained from the environment. The other property, context hiding, is a privacy property. It guarantees that a signature does not reveal anything the message does not already reveal. In particular, if the signature is for m' and derived from a signature for m, the signature should not reveal anything about m. We note that we do not require our constructed signatures to be context hiding, which allows our constructions to be more efficient than constructions that have to be context hiding.

In Section 1.1.1, we dive into our contribution to quotable signatures, i.e., Chapter 2/[BELN23]. Our work on quotable signatures build on what Ahn et al. refers to as a "folklore solution" [ABC+15, p. 8], and which [KNSS19] proposes using to mitigate misinformation. We contribute with a new, more precise performance analysis of folklore solution, concrete algorithms, as well as proving the security of this construction with respect to our (new) notion of unforgeability, which matches the general unforgeability notion for slightly homomorphic signatures from [ABC+15].

While working on quotable signatures for text, we observed that the use of images (and other forms of multimedia content) to spread misinformation, were perhaps more concerning than the use of text. Partly, this is due to images being particularly good vectors for spreading misinformation, due to provoking emotional responses [BPBT06; PBN+23] and – as the idiom goes – one picture being worth a thousand words. The issue with images being used to spread misinformation, was made even pressing with the public release of generative AI models capable of creating pictures that at first glance appear convincing [Bor22]. Thus, we started focusing on approaches to verifying the authenticity of images. Naturally, slightly homomorphic signatures were from the start part of our consideration, but we also considered other methods, for example SNARK based approaches and *perceptual hashing* [DHC20], which we describe in more detail in Section 1.1.3. With images, one also has to consider which transformations should be supported, for example cropping, resampling, color corrections, gray scale conversion, or compression. While approaches supporting multiple of these transformations exist [JWL11; NT16; DEH25; DHC20], they all have other drawbacks, either by only supporting very limited versions of the transformations discussed ([JWL11]), by being very inefficient to compute ([NT16; DEH25]), or by having a non-negligible overlap between their false positive rate and their false negative rate ([DHC20]). For our project, compression seemed to be the fundamental transformation we needed to support, since images are usually compressed when they are uploaded to social media, to save both storage and bandwidth. Compression is also a semantically natural operation for a signature scheme to support: when

an image is (moderately) compressed, it does not fundamentally change what the picture shows.

While some image formats supports lossless compression, meaning that it is possible to restore the image to a bit-for-bit identical one after compression, most image formats are usually compressed with lossy compression. Of the most widely used image formats, only PNG is usually compressed with a lossless compression algorithm (PNG only supports lossless compression). Both JPEG and WebP images are usually compressed in a lossy way, despite both supporting lossless compression. Presumably, this is due to lossless compression not being able to reduce file size sufficiently to meet the needs of modern internet uses. Thus, we need a method for deriving a value from an image, in a way where, if the image is compressed, it is still possible to derive the same value, but where it is hard to find an image resulting in the same value, without that image also being a possible compression of the original image. With such a method, one could create a signature scheme allowing image compression by signing the derived value using a traditional digital signature scheme. This is the core of our contribution, which we describe in more detail in Section 1.1.2. Our work here builds on the work by Johnson, Walsh, and Lamb [JWL11], who first worked on digital signatures for images allowing cropping, by (essentially) splitting the image into chunks, building a hash tree over the chunks, and then signing the root of the tree. When cropping the image, the signature should then be updated to also include some internal nodes from the hash tree, allowing one to still obtain the same root. Johnson, Walsh, and Lamb made the observation that when JPEG compression is performed with a parameter set, where every entry is the same power of two, one could consider this to be cropping away the least significant bits, and then it fitted nicely into their framework, and their scheme could easily be extended to support this very limited form of JPEG compression. Inspired by their work, we developed a more efficient signature scheme, supporting a much wider range of JPEG compression, at the cost of only supporting JPEG compression.

1.1.1 *Contribution: Quotable Signatures for Authenticating Shared Quotes*

As we already mentioned, this article builds on [KNSS19], where the authors suggested using the construction we also use to mitigate the effects of fake news, but they did not go into details. Our main contributions are a formal definition of security for quotable signatures, a detailed description of the construction of quotable signatures including concrete algorithms, a proof that the construction is secure, and a thorough performance analysis. Additionally, we also provide an extended discussion on why quotable signatures is a good approach to mitigate misinformation.

In general, we define a quotable signature scheme to be a quadruple $\mathsf{QSS} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Quote}, \mathsf{Verify})$, of four algorithms, where $\mathsf{KeyGen}, \mathsf{Sign}$, and Verify are generally as in regular signature schemes,

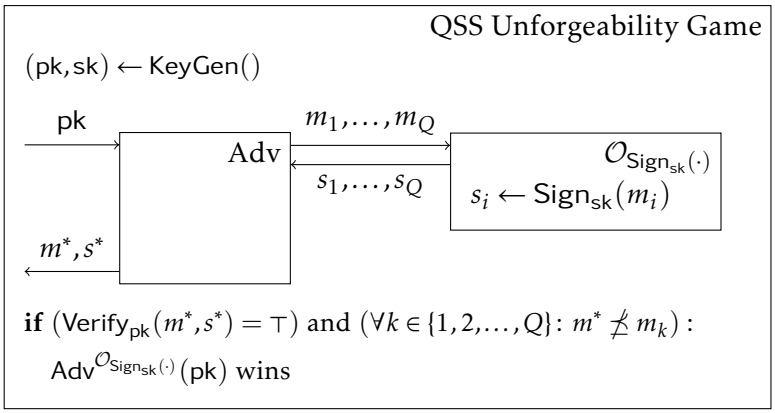


Figure 2: A quotable signature scheme QSS is said to be existentially unforgeable if no adversary wins this game.

but where Quote allows updating a signature σ for m to a signature σ' for m', when m' is a quote from m. We write $m' \preceq m$ to indicate that m' is a quote from m, and hence $m' \not\preceq m$ to indicate that m' cannot be found as a quote in m. Taking inspiration from the relevant literature, we define an unforgeability notion for quotable signatures, by adding a requirement that the message returned from the adversary cannot be contained as a quote in any of the messages it has queried to the signing oracle. We say that a quotable signature scheme QSS is existentially unforgeable if no (PPT) adversary $\mathsf{Adv}^{\mathcal{O}_{\mathsf{Sign}_{\mathsf{sk}}(\cdot)}}(\mathsf{pk})$, wins the QSS unforgeability game in Figure 2 with non-negligible probability.

For our construction of a quotable signature scheme, we build a Merkle tree over a message by first split the message into tokens, which will be the smallest part a quote can either include or exclude, and could for example be words or sentences. If the number of tokens is not a power of two, we additionally require the Merkle tree to be heap shaped. To sign a message, one now uses a traditional signature scheme to sign the root hash of the Merkle tree, instead of a traditional digest of the message. To update a signature σ for a message m to a signature for $m' \preceq m$, one finds the smallest set of internal nodes in the Merkle tree, which together with m' allows finding the root hash of the Merkle tree, while verifying that the tokens of m' are at the claimed positions. This set of tokens is called the *verification path* for m', despite not being on the path between the tokens corresponding to m' and the root. The updated signature σ' for m' consists of σ and the verification path for m'. Note that this procedure can be repeated to find a signature σ'' for $m'' \preceq m'$ from σ'. Verification of a signature is done by first finding the root hash, using the verification path if the message has been quoted from a longer one, and then verifying that the included traditional signature is a valid signature for the root hash. We now state under what conditions this construction is secure and give a short sketch of the proof.

Table 1: Theoretical bounds on the performance of our quotable signature scheme, when quoting t tokens from a text of n tokens.

	Computation Time	Signature Size
The Signer	$2n-1$ hashes and 1 classical signature	1 classical signature
The Quoter		
Arbitrary	$2n-1$ hashes	1 classical signature, at most $t(\lceil \log n \rceil - \lceil \log t \rceil - 1) + 2^{\lceil \log t \rceil}$ hashes
Consecutive	$2n-1$ hashes	1 classical signature, at most $2\lceil \log n \rceil - 2$ hashes
The Verifier	1 classical verification and up to $2n-1$ hashes	—

> Theorem 1.1
> If the Merkle tree is constructed using a cryptographicly secure hash function, and the underlying traditional signature scheme is existentially unforgeable, then $\mathsf{QSS} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Quo}, \mathsf{Ver})$, constructed as outlined above, is an existentially unforgeable quotable signature scheme.

Proof sketch. To prove Theorem 1.1, we argue that any adversary winning the game in Figure 2, implies an adversary against either the underlying traditional signature scheme, or against the hash function used to construct the Merkle tree.

Assume the messages and signatures are labeled as in Figure 2. We first compare the root hash derived from m^* and s^* to the root hashes from $m_1, \ldots, m_Q$. If no root hashes are the same, the adversary has produced a forgery against the underlying traditional signature scheme. If the root hash of m_i is the same, we iteratively go over the nodes in the Merkle trees, starting from the roots, and argue that since $m^* \not\preceq m_i$, we eventually find two nodes with the same value, but with different children. This gives a collision for the hash function. □

After proving that our construction is secure, we analyze the performance of our quotable signature scheme, both the computations required and the signature size. The exact worst case signature size is calculated as a function of the number of tokens quoted and the length of the original message. We pay special attention to the case where the tokens quoted are consecutive, since one could assume this to usually be the case for text quotes – or even enforce that only this form of quoting is allowed. Our results are summarized in Table 1.

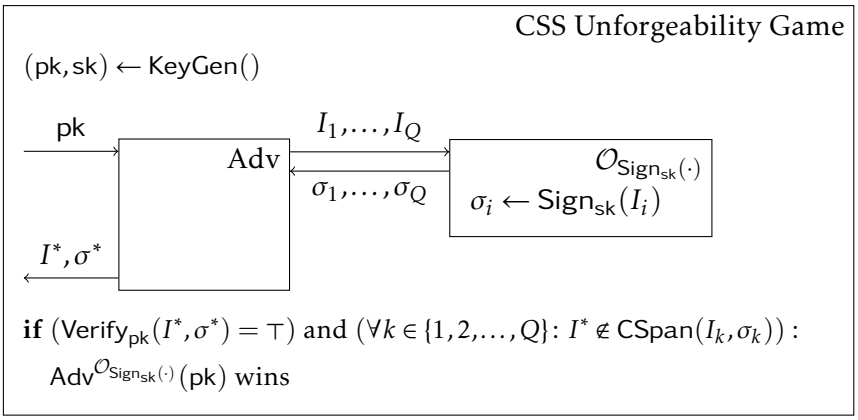


Figure 3: A signature scheme allowing compression CSS is said to be existentially unforgeable if no adversary wins this game.

1.1.2 *Contribution: Digital Signatures for Authenticating Compressed JPEG Images*

The signatures schemes allowing JPEG compression, shares similarities with our work on quotable signatures. We define a signature scheme allowing JPEG compression to be a quadruple of algorithms $\mathsf{CSS} = (\mathsf{KeyGen},\mathsf{Sign},\mathsf{Compress},\mathsf{Verify})$, where all but Compress are similar to a traditional signature scheme. For simplicity, we define the compression algorithm to take as input an image I, a signature σ for I, and parameters P, specifying how compression should be done. Compress should then output I' and σ', where I' is I after compressing it according to P, and σ' is a valid signature for I' signed with the same key as σ. The notion of existentially unforgeability is defined similarly to the CSS definition. First, we defined the *compression span* of an image I with signature σ to be

$$\mathsf{CSpan}(I,\sigma) := \{I' \mid \exists P\colon (I',\sigma') \leftarrow \mathsf{Compress}(I,\sigma,P)\}. \tag{2}$$

Then, we say that CSS is an existentially unforgeable image signature scheme allowing compression, if no (PPT) adversary $\mathsf{Adv}^{\mathcal{O}_{\mathsf{Sign}_{\mathsf{sk}}(\cdot)}}(\mathsf{pk})$, wins the CSS unforgeability game in Figure 3 with non-negligible probability.

Before we describe our construction of a signature scheme allowing JPEG compression, we introduce a key step of the JPEG compression algorithm. The algorithm contains more steps (we describe these in Chapter 3), but the other steps are either optional or lossless, and essentially just converts the image between different representations. The main lossy step of JPEG compression is step 3b, where each 8×8 block of the image, having been converted to a DCT basis in step 3a, is *quantizised* with a *quantization table*. In this step, each value in the 8×8 block is divided by a value from a quantization table, which is an 8×8 table. Why this step is done is not essential for our research, but intuitively, each entry is the coefficient of a DCT wave on the entire 8×8 block, some of which the human vision system is more sensitive to than others, and more information is therefore preserved for the ones we are more sensitive to, by

dividing them with a smaller value before rounding. Two key observations now form the basis for our compression scheme. (1) When a value is divided by a power of two, one can consider it to be *truncation* of the least significant bits of the value (which is represented as either a byte or 10 bits), and (2) the (i,j)'th entry in all 8×8 blocks is quantizised with the same value.[8] Thus, our idea is to create a hash tree structure, where, when coefficients are truncated, one can instead provide internal nodes from the tree, which allows anyone to still verify the remaining part of the coefficients. We use that many coefficients will be truncated by the same amount (observation (2)) to make our construction efficient. Our construction will therefore restrict JPEG compression to only being done with quantization tables with powers of two, but we argue that this still allows a sufficiently high degree of flexibility.

In broad strokes, our construction is as follows: For the (i,j)'th entry, first hash together the least significant bits of all (i,j)'th entries, creating a first (i,j)-digest. Then hash the first digest together with the second least significant bits of all (i,j)'th entries, creating the second (i,j)-digest. Repeat this a total of eight times, obtaining an eighth (i,j)-digest, which is created from the seventh (i,j)-digest and the most significant bits. This process is illustrated for the $(1,1)$'th entries in Figure 4. Finally, hash together the eighth (i,j)-digests for $i,j \in \{1,\dots,8\}$, and sign this digest (called the root hash) with a traditional digital signature scheme. When an image is then compressed with a quantization table where the (i,j)'th entry is 2^ℓ, the ℓ'th (i,j)-digest is added to the signature, allowing anyone to calculate the $\ell+1$'th digest, and so on. This allows the integrity of the bits not truncated to still be verified.

We now state under what conditions this gives an existentially unforgeable signature scheme supporting JPEG compression. We do not provide a sketch of the proof here, but observe that it follows exactly the same approach as the proof sketch for Theorem 1.1.

> THEOREM 1.2
> If the hash tree is constructed using a cryptographicly secure hash function, and the underlying traditional signature scheme is existentially unforgeable, then $\mathsf{CS} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Compress}, \mathsf{Verify})$, constructed as outlined above, is an existentially unforgeable signature scheme supporting JPEG compression with quantization tables containing only powers of two.

Computationally, our scheme is very efficient, requiring at most 1025 hash function evaluations, and one operation from the underlying traditional signature scheme. The size of the signature is also relatively small; just one traditional signature and 128 hash values. As an example, if one were to instantiate the scheme with SHA3-256 and EdDSA using the Ed25519 curve [Dwo15; CMRR23], compressing an image with an initial

8 Technically only the same value as all 8×8 blocks in the same type of channel (luminance or color), but for simplicity we don't go into this here.

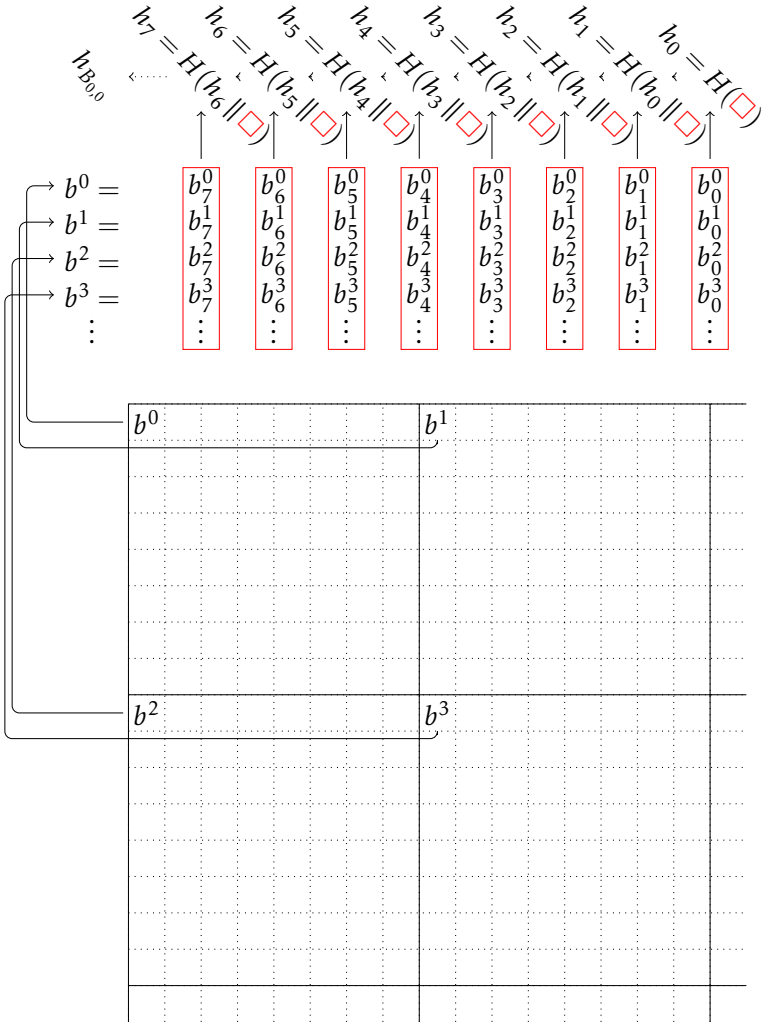


Figure 4: Example illustrating construction of one of the chains of hashes.

Table 2: Average results of compressing the test images from [PLZ+09] with different parameters, compared to the uncompressed images.

		Size	MS-SSIM	FSIMc	MSE	PSNR
QF25	**Our tables**	16.0 kB	0.960	0.978	77.526	29.749
	Unmodified	15.1 kB	0.959	0.978	78.900	29.655
	[JWL11] (64)	10.4 kB	0.914	0.936	109.016	28.021
	[JWL11] (32)	20.5 kB	0.961	0.976	46.071	31.706
QF50	**Our tables**	25.4 kB	0.979	0.991	45.831	32.008
	Unmodified	24.4 kB	0.979	0.991	45.910	31.988
	[JWL11] (32)	20.5 kB	0.961	0.976	46.071	31.706
	[JWL11] (16)	36.5 kB	0.983	0.992	18.451	35.605
QF80	**Our tables**	43.9 kB	0.990	0.997	20.256	35.402
	Unmodified	43.4 kB	0.991	0.997	20.432	35.364
	[JWL11] (16)	36.5 kB	0.983	0.992	18.451	35.605
	[JWL11] (8)	60.4 kB	0.993	0.997	7.532	39.439

size of 2 megabytes down to just 5% of that size, results in an overhead of just 4% (100 kB compressed image and 32.5 kb signature).

Finally, since we restrict the quantization tables to only contain powers of two, we investigate how this affects the visual fidelity of compressed images, compared to images compressed with standard quantization tables. Specifically, we choose the standard quantization tables for quality factor 25, 50, and 80, and then we found quantization tables containing only powers of two, resulting in compressed images of approximately the same size as the standard quantization tables. We compared the resulting images to the uncompressed image using the MultiScale Structural SIMilarity (MS-SSIM) [WSB03] and Feature SIMilarity (FSIM) [ZZMZ11b] image similarity measures, and found averages over the TID2008 image dataset [PLZ+09]. Both of these similarity measures are supposed to indicate how similar the human vision system finds the images. Therefore, we place more weight on them, rather than on traditional distance measures, such as Mean Squared Error (MSE) and Peak Signal Noise Ratio (PSNR), which we do include for comparison. The results are shown in Table 2, where we also compare our approach to [JWL11], which inspired our work. Since their approach requires the quantization table to consist of only one power of two, we have included both the one resulting in the largest smaller image and the smallest larger image. As the table show, our approach results in images that are on average less than 1 kilobyte larger than the unmodified tables, and with almost identical similarity scores.

1.1.3 *Related Work: Verifying Content After Transformations*

Recalling again that the goal of our work was to develop tools for verifying content after transformations, we will now present literature related directly to this aspect of our work. First, in Section 1.1.3.1 we present literature that also has a signature scheme as its core. Then, in Section 1.1.3.2, we present a newer, promising method for verifying image transformations using zk-SNARKs.

1.1.3.1 *Approaches Using Signature Schemes*

Generally, both signature schemes presented in this section are specialized examples of homomorphic signature schemes, which falls into a larger body of literature on this topic [TDB16; ABC+15]. We already described the general definition of slightly homomorphic signatures, or P-homomorphic signatures [ABC+15] at the beginning of Section 1.1, and mentioned that they unify several different frameworks for signatures supporting some type of transformation, for example arithmetic, quotable, redactable, and transitive signatures [KFM04; ZKMH07; BELN23; SBZ01; JMSW02; MR02; BN02]. However, as is often the case, generic P-homomorphic signatures are generally inefficient, compared to more specialized signature constructions.

Redactable Signature Schemes (RSSs) are closely related to our quotable signature scheme, and were simultaneously introduced in [SBZ01] and [JMSW02]. Where the goal of quotable signature schemes is to be able to verify quotes, when they are removed from the text that was signed, the goal of RSSs is to be able to redact parts of a text, and still be able to authenticate the remaining bit – as the name suggest. RSSs therefore require an additional security property: that a redacted signature does not leak any information about the parts that have been redacted. This is typically done by modifying the signature and, similarly to quotable signatures, this modification does not require knowledge of the key used to generate the signature. It is clear that RSSs are stronger than quotable signatures, since RSSs imply quotable signatures, simply by consider quoting as redacting all the text that is not part of the quote, but quotable signatures do not imply RSSs. For example, our quotable signature scheme leaks both the location and length of the removed text, and an adversary trying to guess the redacted text could check their guesses by verifying if hashing it gives the correct internal nodes. Some RSSs require even stronger privacy notions, for example being unable to tell apart fresh and redacted signatures [BPS17]. The advantage of quotable signature schemes' weaker security definition is that it allows more efficient constructions. For example, RSSs with $O(n)$ performance (asymptotically the same as our quotable signature scheme), usually require $O(n)$ expensive public key cryptography operations [BBD+10; SPB+12], where our quotable signature scheme requires only one expensive operation, and $O(n)$ cheap hash function evaluations. Constructions using just one expensive operation either use many more

cheap operations, or result in larger signatures [HK13]. Finally, early RSS definitions in [JMSW02; SBZ01] have a weaker notion of privacy, but still with a hiding element. These constructions end up only slightly more expensive than ours, but we note the performance of these constructions is not as thoroughly analyzed as our construction. In particular, they do not consider the consecutive case, nor how many tokens are quoted.

Aside from the literature on RSSs for text, there is also a large literature on RSSs for other types of data structure, for example different tree structures [BBD+10; SPB+12; HHZ16]. These relate to our signature scheme allowing JPEG compression, which can be viewed as a digital signature on a tree structure, where subtrees are removed when compressing the image. However, RSSs again require the privacy of the removed subtrees, a property we do not require, resulting in our construction being simpler and more efficient.

Sanitizable Signature Schemes (SSSs) are similar to RSSs, but rather than just allowing text (or subtrees) to be redacted, SSSs allow text to be modified. SSSs usually only allow select parts of the text to be modified (which parts should be decided when the original text is signed), and operates with the modification being done by a semi-trusted censor [ACdMT05; BFF+09; dMPPS14; BPS17; BL17]. While one could plausibly construct quotable signatures from SSSs, they suffer from the same efficiency drawbacks due to being capable of more than what is strictly necessary for a quotable signature scheme.

We already discussed the work by Johnson, Walsh, and Lamb [JWL11], who construct signatures for JPEG images, which are homomorphic with respect to cropping, scaling, and (very limited) compression. There have been other works constructing homomorphic signatures specifically for images, but much of this work focused on the JPEG2000 image standard, which was never widely adopted. As of writing, the only web browser supporting the JPEG2000 image standard is Safari [Ado]. An overview of homomorphic signatures for JPEG2000 images can be found in [ZSL04].

An ideal version of our approach to authenticating images would use a "magic" hash function, which for images "looking the same" resulted in the same value, which could then be signed using a traditional signature scheme [Kor17]. This property is exactly the property that *perceptual hash functions* or *robust hash functions* attempt to have [MH80; DHC20]. The issue is that our "magic" hash function also needs to behave like a cryptographic hash function, in the sense that even a minor change in what a picture shows should result in a different pseudo random hash value. All perceptual hash functions known to the author have a non-negligible overlap of having false positives and false negatives [DHC20; LVG12; LC01], and generally appear easy to find collisions for, if the design is known [LP24]. It should be noted that perceptual hash functions are typically used for copyright protection, and sharing of banned images, both use cases where a non-negligible error rate might be acceptable. As a concrete example of perceptual hash functions' shortcomings, Lin and Chang [LC01] design a scheme specifically targeting JPEG im-

ages. Despite this, their scheme accepts images that are manipulated in random, non-targeted ways, with a probability between 0.04 and 0.00001 when the image is not compressed, between 0.09 and 0.0002 with a quality factor of 50, and between 0.2 and 0.02 with a quality factor of 20.

1.1.3.2 *The zk-SNARK approach for Images*

In 2016 Naveh and Tromer created "PhotoProof" [NT16], which is a framework for verifying that images and their associated metadata have only been modified in some claimed ways, using zk-SNARKs. While the construction from [NT16] takes around 5 minutes to generate a proof for a 128×128 pixel image, the idea has been improved upon in [DB23; DCB25; DEH25; MVVZ25], and is getting to a point where it is efficient enough that proof generation take tens of seconds rather than minutes, even for high resolution images. In all the works, the proofs are succinct and verification is efficient. The ideal workflow these approaches suggest is as follows. First, when an image is taken, the image and related metadata is signed by the camera (a camera with this feature was recently released in collaboration between C2PA and Leica [Lyo23]). Alternatively, and depending on the specific application, a party could sign the image and metadata considered to be the original. Then, when the image is transformed in some way, a zk-SNARK essentially proving the following statement is also generated: "The prover (i) knows an unedited photo that is properly signed by a C2PA camera, (ii) the metadata on the unedited signed photo is the same as the one attached to the public photo, and (iii) the public photo is the result of applying the claimed edits to the unedited photo."[9] The application displaying the image should then check the proof of this statement, before displaying the image to the user. Compared to our signature scheme allowing compression, even the recent much more efficient constructions are relatively slow, with the fastest construction [DEH25] taking 15 seconds to generate proofs for an 8K image on a laptop, compared to the 1025 hash function evaluations our signature scheme requires.

1.1.4 *Related Work: Mitigating Misinformation*

In Section 1.1.3, we presented work that relates directly to the cryptographic constructions we have developed. We now focus on work relating to the project goal, of mitigating the negative effects of mis- and disinformation.

At the start of Chapter 1, we already presented some of the alternatives to our suggested approach of verifying the authenticity of content. We focused on the negative labeling approaches used by X/Twitter and Facebook, and on the source-reliability rating approach used by NewsGuard Ratings. For both of these approaches, we also argued that they cannot be sufficient on their own [VRA18; DSÁ20; LES+12; SK20]. Re-

9 This is the example given in [DB23].

calling the C2PA project [C2P] that focuses on tracing the provenance of content, we note that in general their approach require trusting the programs used to edit the images to correctly update the provenance credentials. To support use cases where one or more party might not trust all the programs used for handling the content, supporting some versions of slightly homomorphic digital signatures could be an advantage. For example, one could imagine a situation where one is willing to trust the software used by a journalist to edit an image that will accompany a news story, but where this image is then shared across other platforms that one is not willing to trust, yet still need to compress the image for practical reasons. In this example, the issue could be solved by having the editing software used by the journalist sign the image with a signature scheme supporting JPEG compression.

Another approach to labeling is taken by the "Traceable Original Journalistic Content" project from the Swiss Initiative for Media Innovation.[10] In [LDG+22], they design and develop a certification system, for adding trust indicators to journalistic content, helping readers decide if the news they read is trustworthy. Examples of these labels is if the media is independent, which ethical rules they follow, information about the author, perspectives taken into account, if they have used for example local or expert sources, and more. We note that this work is more focused on certifying the qualities of a news story on the media website, and is thus an orthogonal approach to ours. An analog can be that their approach is more akin to certifying a product (for example as fair-trade), where our approach is more akin to making it clear who produced the product.

Considering work that focuses directly on using tools from cryptography to mitigate fake news, a very simple approach is taken by Amoruso et al. in [AJAZ22], where they propose using standard digital signatures on images. Thus, their approach suffers from the drawbacks that motivated us to investigate slightly homomorphic signatures for images. However, their work goes into details on the technical aspects of *how* a digital signature could be shared together with the image on a social network, both from the client and server side. Thus, their work could naturally be combined with slightly homomorphic signature schemes for text and images. Work by Sidnam-Mauch et al. [SIM+22] surrounds both our work and other works such as [KNSS19; AJAZ22], by taking a holistic approach to which features a cryptographic provenance system should have in order to most efficiently mitigate misinformation. They intentionally use the term cryptographic provenance system to mean the full system, including both what users interact with, and the cryptography running the system. They evaluate the advantages and challenges faced by such a system, drawing on literature human-centered computing, usable security, journalism, and cryptography.

10 https://www.media-initiative.ch

1.2 FOLDING SCHEMES WITH PRIVACY PRESERVING SELECTIVE VERIFICATION (CHAPTER 4)

The third and final article making up this thesis diverges somewhat from the first two, which both focus on very concrete constructions of slightly homomorphic signature schemes. This article instead takes an existing primitive (a folding scheme), and extends it with a new property (privacy preserving selective verification). In essence, a folding scheme allows combining multiple statements from the same NP-language into one statement, which intuitively is in the language if and only of all the initial statements are [KST22]. While folding schemes were initially used for Incrementally Verifiable Computation (IVC), where the verifier wishes to verify all statements, folding schemes have recently found new applications, where the initial statements belong to different verifiers, and where each (honest) verifier is only interested in checking that their respective statement is in the language, leading to the development of *folding schemes with selective verification* [RZ23]. We observed that previous constructions did not provide any notion of "privacy", and, as defined, the process used to convince a verifier that their statement is in the language, leaks a statement belonging to a different verifier. Towards resolving this issue, we define what a folding scheme should satisfy in order to be *privacy preserving*, and we show that some existing folding schemes are amenable to be made privacy preserving.

Before describing our definition and constructions in Section 1.2.1, we describe folding schemes [KST22] and folding schemes with selective verification [RZ23] in greater detail. Let $\mathcal{L}$ be an NP-language with relation

$$\mathcal{R} = \{(x, w) \mid w \text{ is proof that } x \in \mathcal{L}\}. \tag{3}$$

A folding scheme $\mathsf{FS} = (\mathsf{Fold}, \mathsf{FoldVerify})$ for $\mathcal{L}$ consists of two algorithms. The first algorithm, Fold, allows folding two, or more, instances (x_1, w_1) and (x_2, w_2) together into one instance (x, w), which should be in $\mathcal{R}$ if and only if (x_1, w_1) and (x_2, w_2) are in $\mathcal{R}$. Importantly, the folded instance is the same size as each of the instances being folded together. Additionally, Fold produces a *folding proof* π, which the second algorithm, FoldVerify, can use to verify that x is the result of folding x_1 and x_2 (we say that x_i was folded into x). FoldVerify does not require knowledge of any witnesses, w_1, w_2 or w. A folding scheme capable of folding two instances together can be bootstrapped to create a folding scheme for any number of instances, since the folded instance is in the same relation and of the same size as the instances being folded, by composing it recursively with itself and letting the folding proof be all the intermediate statements and folding proofs from the underlying folding scheme.

In [RZ23], Ràfols and Zacharakis observed that while the version of folding schemes just described was well suited for IVC, the use of a single folding proof for verifying all included statements was a potential drawback for other applications. More specifically, an application might involve multiple verifiers, each only interested in checking that one state-

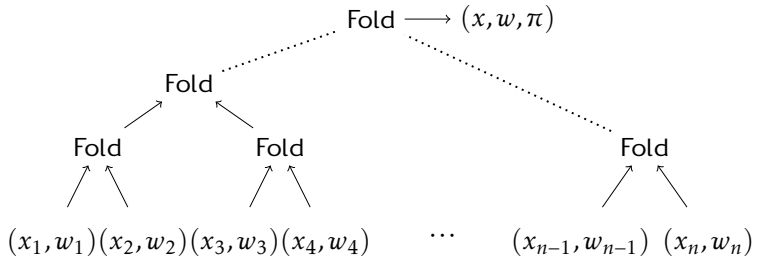


Figure 5: The Merkle tree approach to folding n instances (x_i, w_i) into one instance (x, w), by bootstrapping a 2-folding scheme $\mathsf{FS} = (\mathsf{Fold}, \mathsf{FoldVerify})$.

ment was folded into x. For such applications, it is a drawback that verifying a folding proof requires knowledge of all the statements that were folded together and that the size of the folding proof itself is usually linear in the number of statements folded together. Thus, using the original version of folding schemes might result in unnecessary communication overhead (and some privacy issues, more on those later). Ràfols and Zacharakis resolves this by introducing folding schemes with selective verification, which extends regular folding schemes with two algorithms SlctProve and SlctVerify. These algorithms allow generating and verifying, respectively, individual *selective proofs of folding*, π_i, one for each statement folded together. Each proof of folding only guarantees that the statement it corresponds to was folded into the final statement, and verifying a selective proof of folding only requires knowledge of the corresponding statement. Finally, the size of the selective proofs of folding should be sublinear in the number of statements folded together.

In practice, a folding scheme can be equipped with selective verification in a very straightforward way. When creating a folding scheme capable of folding many instances together by bootstrapping a folding scheme capable of folding two instances together, organize the instances as a binary tree, with the initial instances as the leaves. The intermediate instances are then the internal nodes, and the final instance is the root, see Figure 5. This is similar to a Merkle tree with folding instead of hashing. With this construction, each selective proof of folding π_i consists of the proofs of folding between x_i and the root, together with the statements needed to verify these proofs (in Merkle tree terminology, this would be the verification path of x_i).

The security notions for folding schemes and folding schemes with selective verification are similar, and essentially require two properties: *(selective) completeness* and *(selective) knowledge soundness*. (Selective) completeness guarantees that on valid input, the instance obtained by folding is in $\mathcal{R}$ and FoldVerify is going to accept the (selective) proof of folding. (Selective) knowledge soundness guarantees that if an adversary can produce an instance (x, w) in $\mathcal{R}$ and a valid (selective) proof of folding π which proves that x_i was folded into x, then an extractor can extract a witness w_i for x_i being in $\mathcal{L}$.

1.2.1 *Contribution*

Ràfols presented their work on folding schemes with selective verification at Latincrypt 2023 [RZ23], where I had the pleasure of being in the audience. As a concrete application, Ràfols and Zacharakis suggest delegation of computation, where a folding scheme is used to fold together the proofs that each computation is done correctly. For this use case, selective verification is a natural addition, since each verifier should only want to verify that their specific computation was done correctly. However, for this application, a potential issue with using folding schemes is apparent: to verify a proof of folding, knowledge of the statements folded together is necessary. Using their construction of folding schemes with selective verification does not resolve this issue, since the selective proof of folding includes either the statement before or the statement after the statement the selective proof of folding corresponds to. We resolved this issue by defining and constructing *folding schemes with privacy preserving selective verification*, which we will refer to as *privacy preserving folding schemes*.

Our work first defines what it means for a folding scheme to be privacy preserving. We use an "indistinguishable under chosen message attack"-type definition, where an adversary chooses two different indices i and ℓ, and valid input to the folding scheme, including two different valid inputs for the i'th spot. Then, one of the potential inputs for the i'th spot is chosen at random, and folding is performed. Finally, the adversary is given the result of folding and the ℓ'th selective proof of folding, π_ℓ, and has to figure out which input was used for the i'th spot. This notion of privacy preserving can be thought of as no party being able to learn anything about other parties statements, besides that they are valid (or in other words, that they are in the language).

Towards constructing privacy preserving folding schemes, we construct a new primitive, capable of hiding an instance (x, w) from $\mathcal{R}$ as another instance $(x', w') \in \mathcal{R}$, in a way that (x', w') is in $\mathcal{R}$ if and only if $(x, w) \in \mathcal{R}$, and with the feature that hiding an instance also gives a certificate c, which allows checking that x' hides x. We refer to this primitive as an *NP-statement hider*, and it consists of a hiding algorithm Hide and a checking algorithm Check. The security notion for NP-statement hiders is (almost suspiciously) similar to that of privacy preserving folding schemes. Completeness requires that if $(x, w) \in \mathcal{R}$, then so is (x', w') and the certificate c is accepted by Check. Knowledge soundness requires that if an adversary produces x, c, and (x', w') such that $(x', w') \in \mathcal{R}$ and c is a certificate that x' hides x, then there is an extractor that can extract w from the adversary, such that $(x, w) \in \mathcal{R}$. Finally, the hiding property is also an indistinguishability under chosen message attack notion. An adversary is allowed to choose two instances of $\mathcal{R}$, one of them is hidden, and the adversary is given the hiding instances (x', w') (but not the certificate c) and has to guess which of the instances (x', w') is hiding.

With both an NP-statement hider and a folding scheme with selective verification for a language, constructing a privacy preserving folding scheme is straightforward. First, all instances (x_i, w_i) are hidden as (x'_i, w'_i) using the statement hider. Then, the hidden instances (x'_i, w'_i) are folded together into (x, w), using a regular folding scheme, and selective proofs of folding π_i are generated. Finally, each selective proof of folding π_i is updated to also include the certificate c_i and x'_i, i.e., the statement hiding x_i. Selective verification is now both verifying that c_i is a certificate that x'_i is hiding x_i, and verifying that π_i is a proof of folding showing that x'_i was folded into x.

Thus, what remains is to construct an NP-statement hider. As hinted at when pointing out the similarities in the security notions, we made the very convenient observation that an instance can be hidden using a regular folding scheme with an additional property, which we will describe in a moment. To see how a folding scheme can be used to construct an NP-statement hider, consider that we want to hide $(x_1, w_1) \in \mathcal{R}$. We then sample a random instance $(x_\$, w_\$) \in \mathcal{R}$, and fold the two instances together to obtain $(x, w) \in \mathcal{R}$ and proof of folding π. The instance hiding (x_1, w_1) is (x, w) and the certificate is $c = (x_\$, \pi)$. That this is a secure NP-statement hider will follow from the additional property we will require of the folding scheme. Recall that for the hiding property, an adversary chooses two instances $(x_1, w_1), (x_2, w_2) \in \mathcal{R}$, and receives back (x, w), hiding one of them, and has to figure out which one. Suppose (without loss of generality) that (x, w) hides (x_1, w_1), hidden using random instance $(x_\$, w_\$)$. Then, the additional property that we will require the folding scheme to have is that there is an instance $(x'_\$, w'_\$)$ such that folding (x_2, w_2) and $(x'_\$, w'_\$)$ also gives (x, w), and in fact that there are equally many instances that could be used for hiding (x_1, w_1) and (x_2, w_2) which would result in (x, w). Since the adversary does not learn c, and $(x_\$, w_\$)$ was sampled at random (and hence equally likely to have been $(x'_\$, w'_\$)$), the adversary can do no better than guessing at random.

With the constructions from the last two paragraphs in mind, we state an informal version of the main result of our paper as Theorem 1.3. The proof of the theorem is essentially consists of reducing the security of the privacy preserving folding scheme to the security of the NP-statement hider, and then arguing along the lines of the previous paragraph for security of the NP-statement hider.

> THEOREM 1.3
> Let $\mathcal{L}$ be an NP-language with relation $\mathcal{R}$. If there is a folding scheme for $\mathcal{L}$, $\mathcal{R}$ supports efficient sampling of instances, and for any three instances $(x_1, v_1), (x_2, v_2), (x, v) \in \mathcal{R}$ there are as many instances that fold (x_1, v_1) into (x, v) as there are instances folding (x_2, v_2) into (x, v), then there is a privacy preserving folding scheme for $\mathcal{L}$.

The final contribution of our paper is to show that some languages satisfy the conditions of Theorem 1.3. We do so for both *Inner Product Relation of Committed Values* and *Committed Relaxed C1RS*.

1.2.2 *Related Work*

The main works our construction builds on are the papers mentioned at the start of Section 1.2, namely the paper by Kothapalli, Setty, and Tzialla from CRYPTO'22 which introduced folding schemes [KST22], and the paper by Ràfols and Zacharakis from Latincrypt'23 which introduced folding schemes with selective verification [RZ23]. There are numerous other papers relating to folding schemes, in particular for IVC. IVC is, as the name Incrementally Verifiable Computing suggests, a method for doing verifiable computing, where the computation is done in increments that can each be verified separately [Val08; WAA+24]. Historically, IVC uses *succinct non-interactive arguments of knowledge* (SNARKs) for verifying that each step was computed correctly. However, over the last few years, SNARKs have been replaced with *accumulators* [BGH19; BCMS20; BDFG21; BCL+21], and, most recently, folding schemes, which can be seen as a special, more efficient version of accumulators [NDC+24]. For the traditional SNARK based construction of IVC, the prover constructs a recursive SNARK for each step, each of which prove both that the step was applied correctly to the output of the previous step, and that the SNARK verifier represented as a circuit accepts the SNARK from the previous step [Val08; GGPR13; GW11; BCTV17; BCCT13]. However, this construction is impractical [BCTV17; CCDW20]. One approach to solving this problem is to use a trusted setup for the SNARKs, but even aside from requiring the trusted setup, this has its own efficiency drawbacks, in the form of inefficient verifiers [BFS20; Set20].

A more practical solution is to use accumulators [BGH19; BCMS20; BDFG21; BCL+21]. Where the traditional approach requires verifying a SNARK at each step, accumulators allow deferring the expensive steps of verification, by accumulating these steps together into one instance. At a later time, the accumulated instance can be checked efficiently, and a proof that it is well-formed can also be verified. The drawback of using accumulators is that a SNARK still has to be formed and partially checked at each step. This is where folding schemes come in. Intuitively, folding schemes avoid the multiple SNARKs all together, by folding the *instances* rather than accumulating the *verification*. Folding schemes were first introduced as part of the IVC construction NOVA [KST22], but have since received much interest, and been further researched in SuperNova/HyperNova/NeutronNova [KS22; KS24a; KS24b] (generalizations of NOVA to support non-uniform IVC and the CCS constraint system or the zero-check relation instead of just R1CS), Protostar [BC23a] (targeting special-sound protocols), LatticeFold/Lova [BC24; FKNP24] (post-quantum secure folding scheme), Mangrove [NDC+24] (numerous optimizations to statement generation and IVC construction), BaseFold [ZCF24] (constructing polynomial commitment schemes from folding techniques), and FLI [GM24] (folding scheme for lookup arguments, i.e., convincing a verifier that a set of values appear in a lookup table).

Recent work by Kothapalli (one of the authors who introduced folding schemes [KST22]) and Parno introduces *Reductions of Knowledge* [KP23; Kot24], a framework which generalizes many flavors of arguments of knowledge, including folding schemes. At a high level, reductions of knowledge generalizes schemes that can be thought of as reducing checking knowledge of a witness w_1 for one statement x_1 to checking knowledge of a witness w_2 for a different statement x_2, potentially from a different (usually simpler) relation. Both folding schemes and our new primitive NP-statements hiders can be thought of as reductions of knowledge. NP-statement hiders are reductions from a relation to itself, with an additional hiding property guaranteeing that x_1 cannot be deduced from (x_2, w_2). A folding scheme for a relation $\mathcal{R}$ is a reduction from $\mathcal{R} \times \mathcal{R}$ to $\mathcal{R}$, in the sense that knowing witness (w_1, w_2) for $(x_1, x_2) \in \mathcal{L} \times \mathcal{L}$ is reduced to knowing witness w for $x \in \mathcal{L}$. Unlike folding schemes and NP-statement hiders, reductions of knowledge do not have proofs of folding or certificates for hiding. Rather, they have a property called *publicly reducible*, which requires that given the initial statement and the transcript of the reduction, any party can derive the final statement. This encapsulates the proofs of folding and certificates of hiding for the folding schemes and NP-statement hiders we present, where the proof/certificate is the first message from the prover and potentially part of the input, and verification/checking is exactly reconstructing the final statement.

We already described folding schemes with selective verification [RZ23], and how our work extends the selective verification to be privacy preserving. Related to folding schemes with selective verification, the polynomial commitment scheme hbPolyCommit from [YLF+22] amortizes the cost of batch processing multiple inner-product arguments, corresponding to multiple verifiers, using a Merkle tree structure, similar to how folding is done to support selective verification. This scheme combines multiple protocol transcripts in the Merkle tree structure, allowing each verifier to verify that their transcript was considered at a cost that is logarithmic in the number of transcripts. A similar concept is proposed in [ZXH+22], where they also propose a scheme that can batch prove polynomial commitments. Note that both of these works are different from folding schemes with selective verification, where it is the statements that are aggregated into one statement which is then proven, rather than the proofs being aggregated – in thise sense they are closer to accumulators.

Related to the multi-verifier setting that we operate in are *multi-verifier zero-knowledge proofs* [ACF02; YW22], where they study how to efficiently prove knowledge of the same witness to many verifiers, by allowing the verifiers to communicate. In particular, [YW22] develops proof systems that performance-wise fall between non-interactive zero-knowledge proofs (that can be sent to any number of verifiers, but are comparatively more expensive to generate in the first place, requiring both more memory and computation) and designated-verifier zero-knowledge proofs (which are interactive protocols, but can obtain a

much higher throughput than non-interactive zero-knowledge proofs). The difference between [YW22] and our work is that their goal is to convince all verifiers of the same statement, whereas our goal is to convince each verifier of a different statement. Similarly, a duality in the form of proving many (different) statements to one verifier is *batch zero-knowledge proofs* [BGR98b; BGR98a; HG13]. Finally, one can also consider a duality where there are many different witnesses for the same statement, and the verifier should not be able to tell which one was used [FS90].

Finally, folding schemes have also found another use, tying folding schemes back into the overall theme of this thesis. As we discussed in Section 1.1.3.2, SNARKs can be used to prove that an image has only been transformed in some particular ways, e.g., compressed, resized, had its contrast adjusted, or been converted to gray-scale. The idea of using SNARKs to prove this was suggested in [NT16], but their construction was not practical, with proving times of around five minutes real time for 128×128 pixel images. Since then, improvements were made in [DB23; DCB25; MVVZ25], and recently a new framework VIMz [DEH25] has been proposed. By utilizing folding schemes, VIMz is able to generate proofs for many common transformations of 8K images in fewer than 15 seconds on a regular laptop, demonstrating a real-world application of folding schemes. It is possible that privacy preserving folding schemes could be applied here, enabling further speed-ups by batch-processing multiple images. In comparison to VIMz, our slightly homomorphic signature scheme allowing JPEG compression, discussed in Section 1.1.2, would still be significantly faster, since it needs at most 1025 hash function evaluations and one traditional digital signature operation. On the other hand, our signature approach only supports compression, and is therefore much more limited. Very recently, similar folding based ideas for authenticating video edits have been proposed [ZYO+24]. While currently very inefficient, one could be hopeful that speed-ups similar to those achieved for images will appear.

1.3 OTHER CONTRIBUTIONS (APPENDICES A AND B)

As part of my PhD project, I have also been involved in producing two interdisciplinary manuscripts, placing the cryptographic research presented in my first two manuscripts into a wider societal context, and attempting to disseminate my research to a wider audience, ranging from researchers in the social sciences to data journalists and policymakers.

The first manuscript [GE23], produced in collaboration with Marília Gehrke, who was also a member of the Trust and News Authenticity project at the time, was accepted for the Joint Computation+Journalism Symposium and European Data & Computational Journalism Conference 2023. In this work, we give an overview of our quotable signature scheme, presented here in Section 1.1.1, and argue that quotable signatures are a natural fit for the data journalism tool repertoire. Data journal-

ism's theoretical roots trace back to Precision Journalism [Mey02]. From this, data journalism inherited the key features of the scientific method, often starting with setting up a hypothesis to be tested, followed by collecting data, analyzing and visualizing the data, and finally discussing and drawing a conclusion. From this root, data journalism also inherited a need to be transparent about the data it draws on [Geh22; GM17; Mey02]. Considering that this data often comes from information released under Freedom of Information access legislation [Cod15; Rog13], it could be natural to apply quotable signatures to the data sources the journalists draw on, making it easy to verify that (part of) a data set agrees with what the data journalists used.

The second manuscript [EEG25], produced in collaboration with Johanna Eggers and Marília Gehrke, respectively a current project member and former project member, was submitted to the 2025 Cambridge Disinformation Summit, which aims to "convene global thought leaders to discuss research regarding the efficacy of potential interventions to mitigate the harms from disinformation."[11] In this manuscript, we argue that the use of images as journalistic evidence in the age of generative AI is hazardous [EH23; PBN+23; Hau24]. In particular, generative AI's use in creating "deepfakes" is proving to be of particular concern [PD19; WL22], and approaches for mitigating the threat of AI generated images used in visual misinformation with automatic negative labeling [KFL23] are facing both computational challenges [MSLL21] and potential backfire effects [vdMHO23]. We argue that digital signatures for images could serve as a powerful tool for positive labeling [GDB04] in the defense against visual disinformation. However, for digital signatures to be used for this, they need to be able to follow the images, even when they are shared online. Therefore, using a signature scheme like our slightly homomorphic signatures supporting JPEG compression would be necessary.

11 https://www.jbs.cam.ac.uk/events/cambridge-disinformation-summit-2025/

2

QUOTABLE SIGNATURES FOR AUTHENTICATING SHARED QUOTES

Joan Boyar, Simon Erfurth, Kim S. Larsen, and Ruben Niederhagen. Quotable signatures for authenticating shared quotes. In *Progress in Cryptology - LATINCRYPT 2023*, volume 14168 of *Lecture Notes in Computer Science*, pages 273–292. Springer, 2023. DOI: 10.1007/978-3-031-44469-2_14

ABSTRACT Quotable signature schemes are digital signature schemes with the additional property that from the signature for a message, any party can extract signatures for (allowable) quotes from the message, without knowing the secret key or interacting with the signer of the original message. Crucially, the extracted signatures are still signed with the original secret key. We define a notion of security for quotable signature schemes and construct a concrete example of a quotable signature scheme, using Merkle trees and classical digital signature schemes. The scheme is shown to be secure, with respect to the aforementioned notion of security. Additionally, we prove bounds on the complexity of the constructed scheme and provide algorithms for signing, quoting, and verifying. Finally, concrete use cases of quotable signatures are considered, using them to combat misinformation by bolstering authentic content on social media. We consider both how quotable signatures can be used, and why using them could help mitigate the effects of fake news.

2.1 INTRODUCTION

Digital signature schemes are a classical and widely used tool in modern cryptography (the canonical reference is [DH76], and [CMRR23] contains some current standards). A somewhat newer concept is *quotable signature schemes* [KNSS19], which are digital signature schemes with the additional property that signatures are *quotable* in the following sense. The *Signer* of a message m generates a quotable signature s for m using a private key sk. Given a message m and the quotable signature s, a *Quoter* (any third party) can extract a second quotable signature s' for a quote q from m without knowing sk or interacting with the original *Signer*. A quote can be any "allowable subsequence" of m. We write $q \preceq m$ to indicate that q is a quote from m. This quotable signature s' is still signed with the private key sk of the *Signer* and hence authenticates the original *Signer* as the author of the quote. These signatures for quotes have the same required properties with respect to verification and security as a standard digital signature, in addition to allowing one to derive where

content has been removed, relative to the quote. A signature for a quote is again a quotable signature with respect to sub-quotes of the quote, and neither authenticating a quote nor sub-quoting require access to the original message.

Quotable signatures can be used to mitigate the effects of fake news and disinformation. These are not new problems, and it is becoming increasingly apparent that they are posing a threat for democracy and for society. There is not one single reason for this, but one reason among many is a fundamental change in how news is consumed: a transition is happening, where explicit news products such as printed newspapers and evening news programs are still consumed, but are increasingly giving way for shorter formats and snippets of news on social media platforms [NFKN19]. However, people tend to be unable to recall from which news brand a story originated when they were exposed to it on social media [KFN18]. This is problematic since the news media's image is an important heuristic when people evaluate the quality of a news story [US14]. In addition, according to the Reuters Institute Digital News Report 2022 [NFR+22], across markets, 54% of those surveyed say they worry about identifying the difference between what is real and fake on the Internet when it comes to news, but people who say they mainly use social media as a source of news are more worried (61%).

In recent years, a common approach to fighting back against fake news has been flagging (potentially) fake news, using either manual or automatic detection systems. While this might be a natural approach, research has shown repeatedly that flagging problematic content tends to have the opposite result, i.e., it increases the negative effects of fake news [DSÁ20; LES+12; SK20]. This indicates that flagging problematic content is not sufficient and alternative approaches need to be developed.

We present a method that complements flagging problematic content with the goal of mitigating the effect of fake news. Our idea builds on the observation that *which* news media published a news article is an important heuristic people use to evaluate the quality of the article [US14]. However, since people get their news increasingly via social media, it is becoming more likely that they are not aware of who published the news they are consuming. To address this, we propose using quotable signatures to allow people on social media to find out and be certain of where the text they are reading originates from, and to verify that any modifications to the text were all allowed. Specifically for news, the proposed idea is that a news media publishing an article also publishes a quotable signature for the article signed with their private key. When someone shares a quote from the article, they then also include the signature for the quote that is derived from the initial signature (without access to the private key), which we emphasize is signed with the same key. Finally, when one reads the quote, the signature can be checked, and it can be verified from where the quote originates.

The idea of mitigating the effects of fake news and misinformation, by using digital signatures to verify the source of media content, is one

that has been addressed by others. One example is C2PA [C2P], which involves many companies, including Adobe, the BBC, Microsoft, and Twitter. C2PA focuses on providing a history of a published item, i.e., which device was used to capture it, how it has been edited and by whom, etc. Thus, quotable signatures could be of interest to their approach.

Another issue involving fake news is that news articles are perceived as more credible if they contain attributed quotes [Sun98]. This is misused by fake news to appear more credible by providing attributions for their content [CKA; Sch; Kri; Reu; Dom; HF], but can in turn be used to automatically detect fake news by considering the existence and quality of attributions [AAE+21; TSGS19; MSLL21] (among other things). Quotable signatures, in contrast, could be used to sign quotes to make a strong and verifiable connection between the original source and the quote. On the other hand, fake news would generally not be able to link their quotes to reputable sources, thereby providing another heuristic helping users to distinguish between authentic and fake content.

Without major changes to the system, it could be extended to further use cases such as signing Facebook and Twitter posts, official governmental rules and regulations, scientific publications, etc. For all of these instances, an important feature of our system that we have not used explicitly so far is that signing also binds the Signer, meaning that the signing party cannot later deny having signed the signed document.

We provide an overview over related work in Section 2.2. In Section 2.3, we give a more thorough introduction to and definition of quotable signatures, and we show how we can realize quotable signatures using Merkle trees [Mer80; Mer89]. We define a notion of security for quotable signature schemes, and prove that the notion is satisfied by our construction. Additionally, we prove a number of bounds on the size and computational costs of quotable signatures obtained using Merkle trees. Finishing off the construction of quotable signatures from Merkle trees, we describe algorithms for signing, quoting, and verifying in Section 2.4. We revisit the application of quotable signatures to counter fake news in more detail in Section 2.5 and we conclude the paper with an outlook to future work in Section 2.6.

2.2 RELATED WORK

Quotable signatures have been introduced in [KNSS19], which suggests constructing quotable signatures using Merkle trees and provides a rudimentary complexity analysis. The authors also suggest using quotable signatures to mitigate the effects of fake news. Compared to [KNSS19], we define a security model, and prove that our construction is secure in this security model. Additionally, we also provide proofs of our claims about the cost of using Merkle trees for quotable signatures, provide concrete algorithms for quotable signatures from Merkle trees, and provide more in-depth considerations for why one could expect this to be a good approach.

A concept closely related to quotable signature schemes is *redactable signature schemes* (RSSs). Simultaneously introduced in [SBZ01] (as *Content Extraction Signatures*) and [JMSW02], RSSs essentially allow an untrusted redactor to remove ("redact") parts of a signed message, without invalidating the signature. Often this requires modifying the signature, but crucially, it is still signed with the original key, despite the redactor not having access to the private key. Thus, quotable signatures share many similarities with RSSs; if one considers a quotation as a redaction of all parts of a text except for the quote, they are conceptually identical. Where quotable signatures and RSSs differ is in the security they must provide. Both signature schemes require a similar notion of unforgeability, but an RSS must also guarantee that the redacted parts remain private. A standard formulation is that an outsider not holding any private keys should "not be able to derive any information about redacted parts of a message", and even stronger requirements, such as transparency or unlinkability, are not uncommon [BPS17]. Quotable signatures have no such privacy requirements, allowing quotable signatures to be faster. In fact, it is worth noting that there are scenarios where RSSs' notion of privacy would be directly harmful to a quotable signature. For instance, RSS would specifically make it impossible to tell if a quote is contiguous or not, something that we consider essential for a quotable signature scheme. To see the value of dropping the privacy requirement, we observe that some RSSs with $O(n)$ performance may have $O(n)$ expensive public key cryptography operations [BBD+10; SPB+12], whereas quotable signatures can be obtained with $O(n)$ (cheap) symmetric cryptographic operations (hashing), and only one expensive public key operation. There are approaches obtaining RSSs using only one expensive operation, but they either require many more cheap operations than quotable signatures do, or they result in considerably larger signatures, for example [HK13]. Early examples of RSSs had a weaker notion of privacy, but still stronger than what we require. They require only hiding of the redacted elements, not their location and number. Examples can be found in [JMSW02; SBZ01]. Their approaches are similar to ours, also using Merkle trees, but we provide rigorous proofs of the claimed performance, and our lack of privacy requirements allows our scheme to be both more efficient and conceptually simpler. One consideration that is very relevant for quotable signatures, but seldom considered elsewhere, is how a quote (redaction) being contiguous will affect the complexity results. In a different setting [DGMS00] considers this question for Merkle trees, but provides no rigorous proof.

Considering the motivating example again, approaches to mitigate the impact of fake news, using either digital signatures or directly rating the source of the content, have been proposed and tried before. One approach, serving as inspiration for our approach, is [AJAZ22]. They use digital signatures to verify the authenticity of images and other forms of multimedia. One drawback of their implementation is that it requires the media to be bit-for-bit identical to the version that was signed. Hence, the

image can for instance not be compressed or resized, and thus their solution is not compatible with many platforms, e.g., Facebook compresses uploaded images, and many news websites resize images for different screen sizes. An example of directly rating the source of content, and flagging trustworthy sources, can be found in "NewsGuard Ratings" (NG), which provides a rating of trustworthiness for news sources. NG adds a flag that indicates if a news source is generally trustworthy (green) or not (red) to websites and outgoing links on websites. This approach has not been widely successful. For example, the study in [AGB+22] shows that NG's labels have "limited average effects on news diet quality and fail to reduce misperceptions". While this is somewhat related to our approach, there are two major differences. (1) NG only flags content that directly links to the source of the content with a URL. In contrast, our digital signature can be attached to any text quote. Hence, NG only adds additional information when it is already straightforward to figure out from where the content originates. Our approach also provides this information where there might otherwise be no clear context. (2) NG focuses on providing a rating for how trustworthy a news source is. This approach is similar to the typical approach of telling people when something might be problematic, which tends to have the opposite result. In contrast, we focus solely on providing and authenticating the source of a quote.

Summing up, the contributions of this paper is as follows. (1) We rigorously define the notion of security that quotable signature schemes must satisfy. (2) We rigorously prove the security of and analyze the complexity of, a quotable signature scheme constructed using Merkle trees. (3) This provides a scheme for quotable signatures that is more efficient than using an RSS for the same purpose. (4) We provide concrete algorithms for quotable signatures using Merkle trees.

2.3 QUOTABLE SIGNATURES

To construct a quotable signature scheme, we follow the approach suggested in [KNSS19] and use a combination of a classical digital signature scheme [DH76] and Merkle trees [Mer80; Mer89].

Before getting into the construction, we summarize the setting of quotable signatures. In Section 2.3.1, we define the security notion that quotable signature schemes should satisfy. Then, in Section 2.3.2, we introduce Merkle trees, in Section 2.3.3 we construct a quotable signature scheme and show it is secure, and finally we analyze the complexity of the scheme in Section 2.3.4.

GENERAL SETTING FOR QUOTABLE SIGNATURES. A quotable signature scheme consists of four efficient algorithms, $\mathsf{QS} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Quo}, \mathsf{Ver})$. These four algorithms are essentially the standard three algorithms from a classical digital signature scheme for key generation, signing, and verification, with the added quoting algorithm Quo. To quote from a message, Quo allows extracting a valid signature for the quote

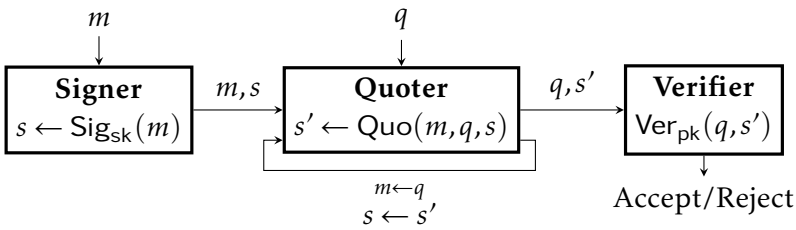


Figure 6: The general setting for a quotable signature.

from the signature of the message in such a way that it is still signed with the public key used to sign the original message. Additionally, it should be possible to derive from the signature of a quote where tokens from the original message have been removed relative to the quote.

We refer to the involved parties as the *Signer*, the *Quoter*, and the *Verifier*. We use λ to denote the security parameter. To summarize:

- $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ takes as input the security parameter 1^λ. It outputs a public key pair. This is typically done by the Signer once, offline as part of the initial setup.
- $s \leftarrow \mathsf{Sig}_{\mathsf{sk}}(m)$ takes as input a secret key sk and a message m. It outputs a quotable signature for m. This is done by the Signer.
- $s' \leftarrow \mathsf{Quo}(m, q, s)$ takes as input a message m, a quote q from m, and a quotable signature s for m. It outputs a quotable signature s' for q, that is still signed with the secret key used to generate s. Verifying s' does not require knowing m. Note that m and s could have been obtained via an earlier quote operation. This is done by the Quoter.
- $\top / \bot \leftarrow \mathsf{Ver}_{\mathsf{pk}}(q, s')$ takes as input a public key pk, a quote (message) q, and a signature s' for q. It outputs $\top$ if s' is a valid signature for q with respect to pk, and $\bot$ otherwise. This is done by the Verifier.

Figure 6 illustrates the typical interactions between the parties.

2.3.1 *Security Model*

Taking inspiration from the RSS notion of unforgeability, we define the security notion of quotable signatures schemes in Definition 2.1. At its core, this is the standard notion of unforgeability for digital signature schemes, with the additional requirement that the adversary's chosen message cannot be a quote from any of the messages that the adversary sent to the signing oracle.

> DEFINITION 2.1 *Unforgeability.*
> Let $\mathsf{QS} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Quo}, \mathsf{Ver})$ be a quotable signature scheme. We say that QS is *existentially unforgeable*, if for every probabilistic polynomial time adversary $\mathcal{A}$, the probability of the following experiment returning 1 is negligible:

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
$(m^*, s^*) \leftarrow \mathcal{A}^{\mathsf{Sign}_{\mathsf{sk}}(\cdot)}(\mathsf{pk})$
// denote the queries that $\mathcal{A}$ make to the signing oracle by $m_1, m_2, \ldots, m_Q$.
if $(\mathsf{Ver}_{\mathsf{pk}}(m^*, s^*) = \top) \wedge (\forall k \in \{1, 2, \ldots, Q\} \colon m^* \not\equiv m_k)$
return 1

2.3.2 *Merkle Trees*

A Merkle tree (also known as a *hash tree*) allows one to efficiently and securely verify that one or more *tokens* are contained in a longer sequence of tokens, without having to store the entire sequence [Mer80; Mer89]. Examples of this could be words forming a sentence, sentences forming an article, or data blocks making up a file.

Since our scheme will rely on hash functions, we assume that the tokens are binary strings. Equivalently, one could assume an implicitly used, well defined injective mapping from the token space to the space of binary strings. For data blocks, the identity mapping suffices and for words one such mapping could be the mapping of words to their UTF-8 representations.

The structure of a Merkle tree for a sequence of tokens is a binary tree, where each leaf corresponds to a token from the sequence, with the leftmost leaf corresponding to the first token, its sibling corresponding to the second token, and so on. Each leaf is labeled with the hash of its token and each internal node is labeled with the hash of the concatenation of the labels of its children. Hence, the i'th internal node on the j'th level will be labeled as

$$u_{j,i} = H(u_{j+1,2i} \,\|\, u_{j+1,2i+1}). \tag{4}$$

This way, one can show that any specific token is in the sequence by providing the "missing" hashes needed to calculate the hashes on the path from the leaf corresponding to the token to the root of the tree. Following established terminology, we call this the *verification path* for the token.[1]

Figure 7 shows the Merkle tree for a sequence of words forming the sentence "The quick brown fox jumps over the dog". The verification path for the word "jumps" consisting of nodes $u_{3,5}$, $u_{2,3}$, and $u_{1,0}$ is highlighted in red. Similarly, one can obtain the verification path for a subsequence of more than just one token. In Figure 7, we also indicate the verification path for the contiguous subsequence "the quick" in blue. Note that the size of the verification path depends not only on how many tokens are chosen, but also on where in the sequence they are placed. In Section 2.3.4, we analyze how large the verification path can become, i.e., how many nodes need to be provided in the signature in the worst case.

1 This use of "path" is slightly counter intuitive, since it refers to the hashes needed to calculate the hashes on the path from the leaf to the root, and hence not the nodes on this path but their siblings.

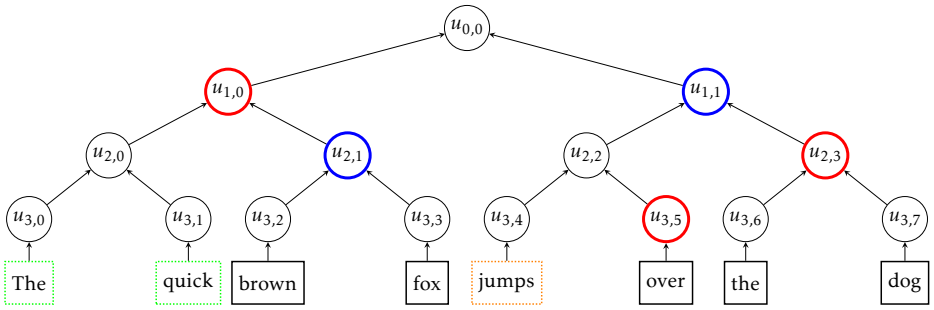


Figure 7: An example of a Merkle tree where the tokens are words and the sequence is a sentence. The verification path for the token "jumps" is highlighted in red $(u_{1,0}, u_{2,3}, u_{3,5})$, and the verification path for the subsequence "The quick" is highlighted in blue $(u_{1,1}, u_{2,1})$.

In these examples, we have chosen a sequence of tokens where the length of the sequence, i.e., the number of tokens, is a power of two. If the sequence length is not a power of two, we require that the tree is *heap-shaped*, i.e., all levels are filled, except for possibly the lowest level, which is filled from the left up to some point, after which the lowest level is empty.

> Remark 2.2
> Observe that from the structure of the Merkle tree, one can see where in the sequence the quoted tokens are placed, and if they are sequential or discontinuous.

2.3.3 *A Quotable Signature Scheme*

Using a Merkle tree, we can now devise a scheme by which the Quoter can convince the Verifier that some quote is contained in a larger text, if the Verifier is already in possession of the root hash. The Quoter simply shares the verification path together with the quote, and the Verifier verifies that this indeed leads to the original root hash. In order to turn this into a quotable signature scheme, we include a classical digital signature for the root hash, signed by the Signer, with the verification path. Thus, letting $\mathsf{DS} = (\mathsf{KeyGen}^{\mathsf{DS}}, \mathsf{Sign}^{\mathsf{DS}}, \mathsf{Ver}^{\mathsf{DS}})$ be a classical digital signature scheme, our quotable signature scheme can be described as follows:

- KeyGen: Identical to $\mathsf{KeyGen}^{\mathsf{DS}}$.
- Sign: Find the root hash of the Merkle tree and sign it with $\mathsf{Sign}^{\mathsf{DS}}$.
- Quo: Find the verification path of the quote. Together with the signature of the root hash, this forms the signature for the quote.
- Find the root hash of the Merkle tree using the quote and its verification path. Use $\mathsf{Ver}^{\mathsf{DS}}$ to verify the authenticity of the root hash.

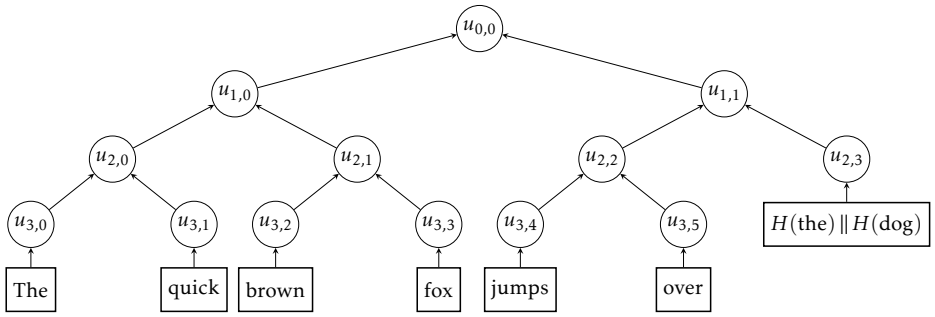


Figure 8: A Merkle tree for the sequence "The quick brown fox jumps over $H(\text{the}) \| H(\text{dog})$", which is a second preimage to the Merkle tree for the sequence "The quick brown fox jumps over the dog".

2.3.3.1 *Proof of Security*

We will show that the construction of the previous section is secure with respect to the notion of security introduced in Definition 2.1, when instantiated with a secure hash function and a secure classical signature scheme. Before doing so, we observe that currently, our scheme is trivially vulnerable to a forgery attack, as follows. An adversary obtains a quotable signature for a message from a signing oracle and then simply replaces the last two tokens on the lowest level with a single token, which is the concatenation of the tokens' hashes. We illustrate this in Figure 8, where we have created a second preimage of the message used in Figure 7. However, there is an easy fix to this vulnerability. Noting that the problem is that an adversary can claim that an internal node is a leaf, we can prevent this by applying domain separation in the form of adding one value to the leaves before hashing, and another value to the internal nodes before hashing. Taking inspiration from RFC 6962 [LLK13], the Merkle trees are modified by prepending 00 to the leaves before hashing and 01 to the internal nodes before hashing. From now on, we implicitly assume that this is done.

We can now argue that the construction is secure.

> THEOREM 2.3
> Under the assumption that
>
> - H comes from a family of cryptographic hash function,
> - $\mathsf{DS} = (\mathsf{KeyGen}^{\mathsf{DS}}, \mathsf{Sign}^{\mathsf{DS}}, \mathsf{Ver}^{\mathsf{DS}})$ is an existentially unforgeable classical signature scheme,
>
> $\mathsf{QS} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Quo}, \mathsf{Ver})$ constructed as described above, is an existentially unforgeable quotable signature scheme.

We have to show that no probabilistic polynomial time adversary can win the unforgeability experiment in Definition 2.1 with non-negligible probability.

Proof. Assume that $\mathcal{A}$ is a probabilistic polynomial time adversary against the unforgeability of QS. We show that the probability of $\mathcal{A}$ being successful is negligible. Let (m^*, s^*) be the output of $\mathcal{A}$, where $s^* = (\mathsf{Sig}^{\mathsf{DS}}_{\mathsf{sk}}(u^*_{0,0}), \{u^*_{i,j}\})$, i.e., the classical digital signature of the root hash and a (possibly empty) verification path.

Consider first the case where the root hash $u^*_{0,0}$ of m^* (found using $\{u^*_{i,j}\}$) is different from the root hashes of the queries $\mathcal{A}$ made to the signing oracle. In this case, $(u^*_{0,0}, \mathsf{Sig}^{\mathsf{DS}}_{\mathsf{sk}}(u^*_{0,0}))$ is a forgery against DS, and since DS is assumed to be existentially unforgeable, this can only happen with negligible probability. Denote this probability as ϵ_{DS}.

If this is not the case, there must be an m_k, such that the root hash of m_k is $u_{0,0} = u^*_{0,0}$, but $m^* \not\subseteq m_k$. Denote by T^* the tree for m^* (constructed using the verification path, if one is included) and by T the tree for m_k.

Consider first the case where all leaves, corresponding to tokens, in T^* are at a location in the tree, where there is also a leaf, corresponding to a token, in T. Since $m^* \not\subseteq m_k$ there must be tokens a^*, a such that $a^* \in m^*$ and $a \in m_k$ are at the same positions in their respective trees, and $a^* \neq a$. Observe that if $H(00 \,\|\, a^*) = H(00 \,\|\, a)$, we have found a collision to H. If $H(00 \,\|\, a^*) \neq H(00 \,\|\, a)$, let the nodes on the path between the leaf corresponding to a^* and the root of T^* be denoted by $u^*_{i,j_i}, u^*_{i-1,j_{i-1}}, \ldots, u^*_{1,j_1}, u^*_{o,o}$ and the nodes on the path between the leaf corresponding to a and the root of T by $u_{i,j_i}, u_{i-1,j_{i-1}}, \ldots, u_{1,j_1}, u_{o,o}$. Since $u^*_{i,j_i} \neq u_{i,j_i}$ and $u^*_{o,o} = u_{o,o}$, there exists a $0 \leq \ell < j$ such that $u^*_{\ell,j_\ell} = u_{\ell,j_\ell}$ and $u^*_{\ell+1,j_{\ell+1}} \neq u_{\ell,j_{\ell+1}}$. Thus, $u^*_{\ell+1,j_{\ell+1}}$ and $u_{\ell,j_{\ell+1}}$ (together with their siblings and 01) form a collision.

Consider now the case where there is a leaf, corresponding to a token, in T^* that is not at a location in the tree, where there is a leaf, corresponding to a token, in T. In this case there must be nodes $u^*_{i,j} \in T^*$ and $u_{i,j} \in^T$ at the same position in their respective trees such that one of them is internal and the other corresponds to a token. If $u^*_{i,j}$ and $u_{i,j}$ do not have the same label, we can apply the method from the precious paragraph to find a collision. If they have the same label, we must have two nodes $u_{i+1,2j}, u_{i+1,2j+1}$ in T or T^*, and a token a in m^* or m_i such that $H(01 \,\|\, u_{i,j} \,\|\, u_{i,j+1}) = H(00 \,\|\, a)$, and we have found a collision.

We observe that in all cases, we have found a collision for H. Since H is assumed to be secure, and hence collision resistant, this can happen only negligible probability. Denote this probability as ϵ_H.

Hence, $\mathcal{A}$'s advantage of at most $\epsilon_{\mathsf{DS}} + \epsilon_H$ is negligible. □

2.3.4 *Performance*

Table 3 shows the cost of our quotable signature scheme for each of the three parties. This is measured in terms of computation due to the number of required hash operations and classical signature operations as well as in terms of the size of the generated signature due to the required hash values and classical signatures, presumably the dominant operations. In all cases, we assume that the message m has length n, i.e., m consists of n

Table 3: Theoretical bounds on the performance of our version of a quotable signature. For the Quoter, we consider both if we allow quoting arbitrary tokens from the sequence, and when we require that the quoted tokens must be consecutive.

	Computation Time	Signature Size
The Signer	$2n-1$ hashes and 1 classical signature	1 classical signature
The Quoter		
Arbitrary	$2n-1$ hashes	1 classical signature, at most $t(\lceil \log n \rceil - \lceil \log t \rceil - 1) + 2^{\lceil \log t \rceil}$ hashes
Consecutive	$2n-1$ hashes	1 classical signature, at most $2\lceil \log n \rceil - 2$ hashes
The Verifier	1 classical verification and up to $2n-1$ hashes	—

tokens. For the Quoter and the Verifier, we additionally assume that the quote has length $t \leq n$.

To put the results into context, running the command `openssl speed` on a modern laptop shows that it is capable of computing hundreds of thousands or even millions of hashes every second (depending on the size of the data being hashed and the hash algorithm being used). Additionally, a classical digital signature only takes a fraction of a second create or verify. Thus, it is nearly instantaneous to generate/quote/verify a quotable signature, even for sequences and quotes that are thousands of tokens long.

The cost for the Signer, the Quoter, and the Verifier is derived as follows.

2.3.4.1 *The Signer*

Computing the cost for the Signer is straightforward. To generate the Merkle tree, the Signer needs to compute $2n-1$ hashes. To create the quotable digital signature for m, she creates a classical digital signature for the root hash. This classical digital signature is the Signer's signature for her message m.

2.3.4.2 *The Quoter*

The Quoter also has to generate the entire Merkle tree, from which he can extract the verification path for the quote he wishes to make. However, the size of the verification path (and hence the signature for the quote) depends on the size of the quote, and where in the text the quote is located. The most simple case is when just one token is quoted, in which case the size of the verification path is at most $\lceil \log n \rceil$, which, together with the classical signature for the root hash, forms the signature for the quote.

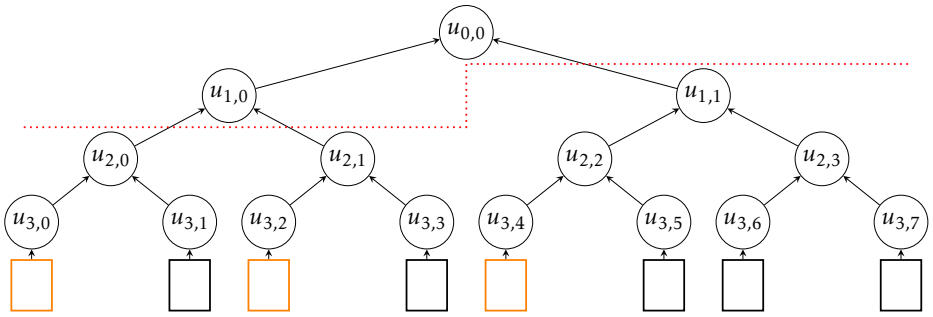


Figure 9: A Merkle tree for a sequence of size $n = 8$ and a quote of size $t = 3$.

Similarly, as shown in the following, the worst case can be obtained by quoting every second token, in which case the Quoter would need $\lceil \frac{n}{2} \rceil$ hashes on the verification path.[2]

In Proposition 2.4 we quantify the worst-case size of the verification path (and hence the signature) for the quote in terms of message and quote lengths. In Proposition 2.10, we consider the special case where we require that the quote be contiguous.

> Proposition 2.4
> For a message m of size n tokens and a quote of size t tokens, the worst-case size of the verification path of the quote is at most
>
> $$t(\lceil \log n \rceil - \lceil \log t \rceil - 1) + 2^{\lceil \log t \rceil}. \tag{5}$$

Proof. In Lemma 2.6, we consider the case where n is a power of two. In this case, we identify a worst-case set of t leaves of the Merkle tree on n tokens. In Lemma 2.7, we establish that it is sufficient to consider n a power of two.

To argue about the size of the signature, we consider what we call the *forest of independent trees* for a quote. To find the forest of independent trees for a quote, we do the following. For each token in the quote, consider the path between the node corresponding to that quote and the root (the root-token path). Define the *independent tree corresponding to that token* to be the subtree rooted in the highest node on the root-token path, which is not on the root-token path for any other token in the quote. The forest of independent trees for the quote is now the collection of the independent trees of all the tokens in the quote. In Figure 9, we consider a message of size $n = 8$ and a quote of size $t = 3$, quoting the first, third, and fifth token. The red line indicates a separation between the independent trees and the nodes that are on multiple root-token paths. The forest of independent trees consists of the trees rooted in $u_{2,0}, u_{2,1}$, and $u_{1,1}$.

2 Of course, algorithms can be adapted to include the entire text instead in such (rare) cases where that might require less space.

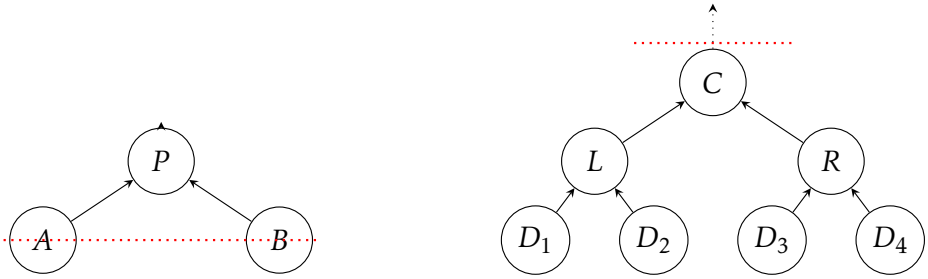


Figure 10: Note that there might be trees rooted at A, B, D_1, D_2, D_3, and D_4, which we have omitted drawing, but by our assumption, the trees rooted at D_1, D_2, D_3, and D_4 must be at least as high as the ones rooted at A and B.

LEMMA 2.5
If n is a power of two, the heights of the trees in the independent forest for a quote that maximizes the size of the signature can differ by at most 1.

Proof. Assume towards a contradiction that Q is a quote that maximizes the size of the signature for Q such that the difference between the heights of the smallest and largest trees in the forest of independent trees for Q is at least 2. Let A be the root of a tree of minimal height in the forest of independent trees, and let B be its sibling. Note that B is also the root of a tree in the forest of independent trees (otherwise the tree rooted at A would not be of minimal height). Additionally, let C be the root of a tree of maximal height in the forest of independent trees. We illustrate this in Figure 10.

Observe that we can now create a quote Q' requiring more hashes than Q, by changing Q in the following ways:

- Instead of quoting one token from the tree rooted at A and one token from the tree rooted at B, Q' quotes only one token from the tree rooted at P.
- Instead of quoting just one token from the tree rooted at C, Q' quotes one token from the tree rooted at L and one token from the tree rooted at R.

It is clear that Q and Q' quote equally many tokens and that the forest of independent trees for Q' is only changed from the forest for Q in the trees that involves A, B, and C. The new situation is illustrated in Figure 11.

If each of the trees rooted at A and B contributed with k hashes to Q, then the tree rooted at C contributed with $k' + 2$ hashes, where $k' \geq k$. In total, A, B, and C contributed $2k + k' + 2$ hashes. However, in Q' we see that the tree rooted at P contributes $k + 1$ hashes, and each of the trees rooted at L and R contributes $k' + 1$ hashes, for a total of $k + 2k' + 3$ hashes. But since $k' \geq k$, we have that

$$k + 2k' + 3 \geq 2k + k' + 3 > 2k + k' + 2, \tag{6}$$

contradicting that Q maximizes the size of the signature. □

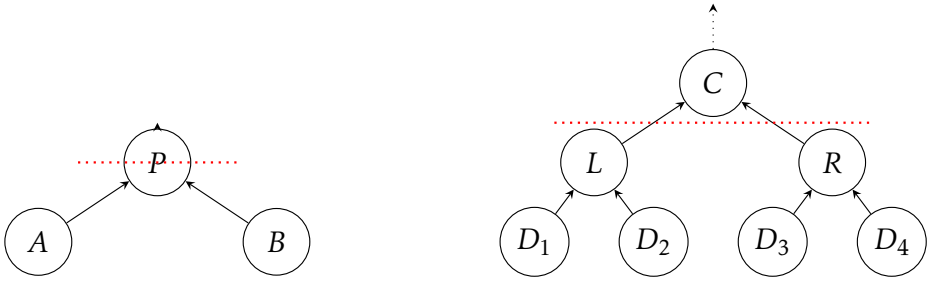


Figure 11: Note that there might be trees rooted at A, B, D_1, D_2, D_3, and D_4, which we have omitted drawing, but by our assumption, the trees rooted at D_1, D_2, D_3, and D_4 must be at least as high as the ones rooted at A and B.

> LEMMA 2.6
> When n is a power of two, we can assume that the quote generating the largest signature has the properties that
>
> 1. the heights of the trees in the independent forest for the quote differ by at most 1,
> 2. for each tree in the forest of independent trees, the left-most leaf corresponds to the token that is quoted, and
> 3. the trees in the forest of independent trees are arranged with the smallest trees first.

Proof. Claim 2.6.1 follows immediately from Lemma 2.5. Further, Claim 2.6.2 follows from observing that we can bring any tree to this form simply by swapping the children of some of the nodes on the path to the leaf corresponding to a quoted token (hereby changing which token is quoted, but not how many are quoted), and that these swaps do not affect the size of the signature. Finally, Claim 2.6.3 follows from observing that if two nodes are on the same level of the Merkle tree, and the labels of both are known, then we can "swap" the subtrees that they are roots of without affecting the size of the signature. By "swapping", we mean that if the i'th leaf in the first node's subtree corresponds to a quote before the swap, then the i'th leaf in the second node's subtree corresponds to a quote after the swap, and so on. To see that this does not affect the size of the quote, note that outside of the two subtrees, nothing has changed; the hash of both nodes is still known. Additionally, from the first subtree we now get as many hashes as we got from the second subtree before the swap, and vice versa. □

Lemma 2.6 implies that for any n a power of two and $t \leq n$, we need only consider one choice of which tokens are quoted. For example, Figure 9 shows the only quote of size $t = 3$ in a tree of size $n = 8$ that we need to consider.

LEMMA 2.7
For any message m of length n and quote Q of length t, there is a quote Q' of length t from a message m' of length $2^{\lceil \log n \rceil}$ such that the signature for Q' is no smaller than the signature for Q.

Proof. For fixed m and Q, we create m' by adding tokens to m until $|m'| = 2^{\lceil \log n \rceil}$. We now create Q' from Q by going over each quote q in Q.

1. If the leaf corresponding to q in the Merkle tree for m is on the deepest level, we quote the same token in m'.
2. If the leaf corresponding to q in the Merkle tree for m is not on the deepest level, there is an internal node in the Merkle tree for m' at the location of the leaf in m. We quote the token corresponding to its left child, which is a leaf.

Clearly, the tokens in Q' from case 1 contribute with the same number of hashes to the signature for Q' as the corresponding ones did to the signature for Q, and the tokens from case 2 contribute with exactly one more hash. Hence, the signature for Q' is at least as large as the signature for Q. □

We are now ready to derive the claim in Proposition 2.4. For any message m and quote Q we can assume that $|m| = n$ is a power of two, i.e., $n = 2^{\lceil \log n \rceil}$ (otherwise Lemma 2.7 allows us to instead consider an m' that is a power of two), and that Q has size $|Q| = t$ and exactly the structure described in Lemma 2.6.

There are t trees in the forest of independent trees for the quote, and all the way up to (but not including) their roots, each of these trees provides one hash per level. The roots of the trees in the forest are on the deepest level with less than t nodes and the first level with more than t nodes (if t is a power of two, all roots are instead on the level with exactly t nodes). Hence, all levels that are at depth more than $\lceil \log t \rceil$ contributes with 1 hash per tree, for a total of $t(\lceil \log n \rceil - \lceil \log t \rceil)$ hashes. Additionally, we need to count how many hashes we get from the level at depth $\lceil \log t \rceil$. On this level, every node is either a root of an independent tree or a child of a root of an independent tree. In the first case, the hash of the node is calculable from information from lower levels. In the second case, for every pair of siblings, one of the nodes' hash is calculable from information from lower levels (the one on a root-token path for a token corresponding to a quoted token) and the other nodes' hash must be provided by the signature. Since there are $2^{\lceil \log t \rceil}$ nodes on this level, and t independent trees, the signature must provide $2^{\lceil \log t \rceil} - t$ hashes on this level.

In total, this shows that an upper bound on the number of hashes provided by the signature for a quote of t tokens from an n tokens sequence is

$$t(\lceil \log n \rceil - \lceil \log t \rceil) + 2^{\lceil \log t \rceil} - t \tag{7}$$

$$= t(\lceil \log n \rceil - \lceil \log t \rceil - 1) + 2^{\lceil \log t \rceil}, \tag{8}$$

which finishes the proof of Proposition 2.4. □

> COROLLARY 2.8
> For a message of size n tokens and any quote, the worst-case size of the verification path of the quote is $\lceil \frac{n}{2} \rceil$.

Another easy corollary to the proof of Proposition 2.4—and Lemma 2.7 in particular—we can bound the error when n is not a power of two (when n *is* a power of two, the bound is, of course, exact).

> COROLLARY 2.9
> When n is not a power of two, the bound of Proposition 2.4 overcounts by at most t hashes.

Proof. At each level of the Merkle tree, the signature needs to provide at most one hash for each quoted token. In the construction used in the proof of Proposition 2.4 when n is not a power of two, no levels are added to the Merkle tree, and hence the signature becomes no more than t hashes larger. □

> PROPOSITION 2.10
> For a message of size $n > 2$ tokens and a contiguous quote of t tokens, the worst-case size of the verification path of the quote is $2\lceil \log n \rceil - 2$ hashes.

Proof. We prove this proposition by induction on the height of the Merkle tree.

As the base case, we consider trees of height 2. Either picking just one token or picking one token among the first two tokens and one token among the last one or two tokens, gives a verification path of worst-case size $2 \cdot 2 - 2 = 2$.

Assume now that in a tree of height k, the largest possible size of the verification path for a contiguous quote is $2k - 2$. As our inductive step, we show that if the height of the Merkle tree of a message is $k+1$, then the largest possible size of the verification path for a contiguous quote from the message is $2(k+1) - 2$. For any contiguous quote Q, we consider two cases: (1) Q is either contained in the first 2^k tokens or contains none of the first 2^k tokens, and (2) Q contains both the 2^k'th and the $(2^k + 1)$'st token.

CASE 1: If Q corresponds to leaves that are completely contained in one of the subtrees of the root, it follows from the induction hypothesis that the verification path consists of at most $2k - 2$ hashes from that subtree. The verification path contains only one additional hash, that of the root of the other subtree. Thus, the total number of hashes is at most $2k - 2 + 1 < 2(k+1) - 2$.

CASE 2: We make a few observations. Considering a level of the Merkle tree from left to right, the nodes with hashes that the Verifier calculates are consecutive. In Figure 12, we have illustrated this by highlighting in green all the nodes with labels that the Verifier calculates.

Additionally, observe that for any level of depth $j \geq 2$, the only nodes of depth $j - 1$ with a label that the Verifier has to calculate and that, at

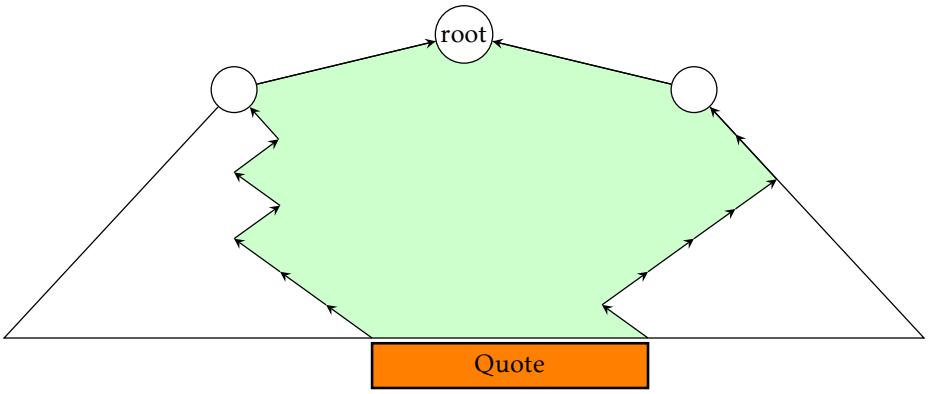


Figure 12: Merkle tree with a contiguous quote divided between the left- and right subtree. The labels of all the nodes in the green area are calculated from the labels of their children and do not need to be part of the signature.

the same time, (potentially) has a child outside the consecutive sequence of nodes that the Verifier calculated the labels for at depth j, are the parents of the leftmost and rightmost nodes in that consecutive sequence at depth j. All the nodes that might be characterized like this are on the two paths of black arrows in Figure 12. Hence, it follows that on each level, the verification path needs to provide at most 2 hashes. Clearly, the root's label will not need to be provided by the verification path, and the root's children will also not need to have their labels provided since the quote contains a token from each child's subtree. Finally, observing that there are a total of $k+2$ levels in a tree of height $k+1$, allows us to conclude that the verification path needs to provide at most $2 \cdot (k+2-2) = 2 \cdot (k+1) - 2$ hashes, completing the case and the proof. □

2.3.4.3 *The Verifier*

The Verifier has to verify one classical digital signature and to reconstruct the Merkle tree using the quote together with the verification path. Once again, the cost of this depends on where in the message the quote is located, with the number of hashes generally going towards $2n-1$ as the quote gets closer to being the full message. For example, if all but one token has been quoted, the Verifier needs to compute $2n-2$ hashes, and if only one token has been quoted, the Verifier only needs to compute $\lceil \log n \rceil + 1$ hashes.

2.4 DESIGN

This section considers some more practical aspects of quotable signatures from Merkle trees. In Section 2.4.1, we give an overview of how the parties' algorithms work. Then, in Section 2.4.2, we discuss some application-specific choices one would have to make when implementing or using quotable signatures.

2.4.1 *Algorithms*

Before going into the details of the algorithms, we consider the intuitive approach one would take on the example shown in Figure 7:

- First, we need an algorithm (described in detail in Section 2.4.1.1) for generating the Merkle tree shown in Figure 7. It takes the sequence of words (the tokens) as input and outputs the label of the root node $u_{0,0}$. The tokens are given to the algorithm by the signing or quoting algorithms. The tree computation is somewhat trivial but important since different algorithms might result in different tree shapes, but the same shape is required for signing, quoting, and verifying. In our case, we require the tree to be heap-shaped.
- The signing algorithm (described in Section 2.4.1.2) extracts the sequence of tokens from the message, applies the algorithm for generating a Merkle tree, and signs the label of the root node.
- The quoting algorithm (described in Section 2.4.1.3) computes the nodes on the verification path of a quote along with their labels. Referring back to Figure 7, this starts with identifying the nodes highlighted in blue, if the quote was the subsequence "The quick", and in red, if the quote was "jumps". This algorithm also generates the Merkle tree and extracts the information needed to verify the quote such as the location of the quote in the original message as well as its length.
- The verifying algorithm (described in Section 2.4.1.4) is given the quote and the quoted signature, which consists of the location of its tokens within the original message, the length of the original message (number of tokens), the labels of the nodes on the corresponding verification path, and the digital signature for the label of the root hash. From this it calculates the label of the root node in the Merkle tree corresponding to the full sequence and verifies that the given signature is indeed a valid signature for this value. If the quote was "The quick", this corresponds to calculating the labels of $u_{3,0}, u_{3,1}, u_{2,0}, u_{1,0}$, and finally $u_{0,0}$, which would then be verified.

The following sections describe these algorithms in detail, beginning with the computation of the Merkle tree, since this operation is required for both signing and quoting.

2.4.1.1 *Generating Merkle Trees*

To generate a Merkle tree for a sequence S of tokens, we define `CreateMerkleTree(`S`)` as a recursive function. We use ℓ to indicate the number of tokens in S.

- If S consists of just one token, then create a new node with the token as its label. Let this be the sole child of a new node u, with the hash of the token as its label. Return u.

- Otherwise, create a new node u.
  - If ℓ is a power of two, then let u's left child be the node returned by recursively calling `CreateMerkleTree()` on the first $\ell/2$ tokens of S, and let u's right child to be the node returned by recursively calling the `CreateMerkleTree()` on the last $\ell/2$ tokens of S.
  - If ℓ is closer to $2^{\lceil \log_2 \ell \rceil}$ than $2^{\lfloor \log_2 \ell \rfloor}$, the tree rooted at u's left child will be full and contain $2^{\lfloor \log_2 \ell \rfloor}$ tokens. Hence, let u's left child be the node returned by recursively calling `CreateMerkleTree()` on the first $2^{\lfloor \log_2 \ell \rfloor}$ tokens of S, and u's right child be the node returned by recursively calling `CreateMerkleTree()` on the remaining $\ell - 2^{\lfloor \log_2 \ell \rfloor}$ tokens of S.
  - If ℓ is closer to $2^{\lfloor \log_2 \ell \rfloor}$ than $2^{\lceil \log_2 \ell \rceil}$, the tree rooted at u's right child will be complete, and contain $2^{\lfloor \log_2 \ell \rfloor - 1}$ tokens. Hence, let u's right child be the node returned by recursively calling this function on the last $2^{\lfloor \log_2 \ell \rfloor - 1}$ tokens of S, and u's left child be the node returned by recursively calling this function on the remaining $\ell - 2^{\lfloor \log_2 \ell \rfloor - 1}$ tokens of S.
- Set u's label to be the hash of the concatenation of the labels of u's children, i.e., $u.\text{label} = \text{hash}(u.\text{left.label} \,\|\, u.\text{right.label})$.[3]
- Return u.

This function returns the root of the Merkle tree corresponding to S. The Signer signs the label of the root to create the digital signature for the message corresponding to S.

2.4.1.2 *Signing a Message*

Using `CreateMerkleTree()`, signing a message is straightforward.

- Turn the message m into a token sequence S. The Quoters and Verifiers need to be able to obtain the same tokens for a given message or quote. How this can be achieved depends on the specific application and use-case; see Section 2.4.2 for a brief discussion.
- Generate the Merkle tree for S using `CreateMerkleTree()`. Denote the label of the root of the Merkle tree by u.
- Sign u using a classical signature algorithm to obtain the quotable signature for the message m.

2.4.1.3 *Quoting a Message*

To obtain a quote Q for a subsequence of the token sequence S, the Quoter does the following:

3 Note that we are omitting that we have to mask the values of tokens and the labels of internal nodes in different ways before hashing them, in order to avoid a trivial collision attack, as discussed in Section 2.3.3.1.

- Extract the token sequence from the message and generate the Merkle tree using `CreateMerkleTree()`.
- Add a flag to each internal node in the created Merkle tree that indicates if the label of the node needs to be provided in the signature for the quote. Initially, set each flag to `delete`, indicating that they are not needed.
- For each token *in the quote*, process each node on its root-token path as described below (start at the node corresponding to the token and, after processing that node, continue to its parent, stopping after finishing with a child of the root). Note that when processing a later token, nodes on its root-token path may no longer have their flag set to `delete` if they have already been processed on another root-token path.
  - If the node's flag is `delete`, set its sibling's flag to `required`, indicating that its label is needed (unless this flag is later changed to `implicit`). Note that this node and its sibling could both correspond to tokens in the quote, in which case, when the sibling is processed, both nodes will have their flags set to `implicit`.
  - If the node's flag is `required`, set the flags of the node and its sibling to `implicit`, indicating that their labels can be calculated from information that is already included. Then move on to the next token; the rest of the verification path for this token has already been considered, as part of the verification path for a previously processed token.
- Extract the hashes that the signature needs to provide by performing an inorder traversal of the Merkle tree, adding the label of any node with its flag set to `required` to a list of provided hashes.
- Create the signature for the quote as the signature for the root of the Merkle tree, the list of hashes generated in the previous step, the number of tokens in the original message, and the indices of the quoted tokens.

> Remark 2.11
> Note that we have made the assumption that the Quoter is quoting directly from a message and not quoting from a quote. However, one can straightforwardly combine the latter parts of this algorithm with parts of the algorithm described in Section 2.4.1.4 to obtain this functionality.

2.4.1.4 *Verifying a Quote*

Given a quote and a signature for the quote, consisting of the signature for the root of the Merkle tree, a list of required hashes, the length of the

original sequence, and the indices of the quoted tokens, the Verifier can verify the quote as follows:

- Create a heap-shaped tree with as many leaves as there were tokens in the original sequence. Let all the nodes be unlabeled.
- Add a flag to each node in the tree, initially setting each flag to `delete`.
- For each token *in the quote*, work upwards on the root-token path corresponding to the token. For each node (except the root), do the following:
  - If the node's flag is `delete`, set its sibling's flag to `required`.
  - If the node's flag is `required`, set the flags of the node and its sibling to `implicit` and move on to the next token.
- Perform an inorder traversal of the tree. When encountering a node with its flag set to `required`, label it with the next hash in the list of required hashes.
- For each of the leaves corresponding to tokens in the quote, label them with the hash of that token.
- The remaining labels, including the root's, can now be calculated using a straightforward recursive function: Starting from the root, calculate its label from the labels of its children, calling recursively on any unlabeled children.
- Verify the calculated root hash, with respect to the signature for the root hash, included in the signature for the quote. If this verification is successful, the quote has been verified.

This only covers verifying the authenticity of the quote, and additional information could be made clearly available. This information could, for example, include if the quote is contiguous or where tokens from the original message are missing, where they were located in the message, and application-specific information.

2.4.2 *Application-Specific Choices*

When instantiating quotable signatures for a concrete use-case or application, one of the choices to make is what to use as tokens. In our examples, we have used words as tokens, which could be a natural choice for some applications, but there are many other ways to tokenize a message. This is considered further in Section 2.5.

For the Signer's algorithm in Section 2.4.1.2, there is a choice to be made as to what classical digital signature scheme is used to sign the root's label. Here, suggestions could be to follow either one of schemes

from the Digital Signature Standard (DSS) [CMRR23], or, in the interest of long-term security, a post-quantum signature scheme such as [DKL+18; FHK+20; ABB+22].

A natural optimization would be to change the Merkle trees to use tokens as leaves instead of hashes of tokens. This would reduce the number of hash calculations needed to construct a Merkle tree by about a half. One can in some situations take this slightly further. If the combined size of the tokens of two leaf children of a node is no longer than a hash value, then we could use the concatenation of the two tokens instead of a hash value for their parent. Naturally, this could be continued recursively.

2.5 QUOTABLE SIGNATURES AND FAKE NEWS

In the introduction, we argued that the current approach to mitigating the effects of fake news, focusing on flagging problematic content, is not sufficient. As mentioned, one supplementary approach could be to bolster authentic content by authenticating the source of quotes, for example on social media, and the literature gives reason to believe this could have an impact. This approach could be implemented using a quotable signature scheme. Here, the message that is the original source of a quote would be an article and the creator or distributor of the article (a news agency, for instance) would act as the Signer, the one sharing the quote as the Quoter, and the one verifying the quote as the Verifier. For this approach to be effective, it would need to be widely adopted, both by news media and by users sharing and reading quotes from articles. We make the following observations on these problems.

Regarding the news media, there is wide interest in supporting initiatives to combat fake news, see for example [C2P]. Additionally, from our discussions with a news media company,[4] it is apparent that the current workflow employed by modern media companies is already highly automated, and it appears that it should be quite simple to integrate a process by which, when an article is published (or updated), it is automatically signed with the media company's public key. Regarding user adoption, there is the challenge of getting a sufficiently large proportion of users using the tool, but one would also have to teach users what a quote being authenticated means, i.e., that the source and integrity of the quote has been assessed, but not its truthfulness or the quality of its source, for example.

If news media and social media integrate this approach into their websites, our algorithms can be employed without any explicit user awareness. With such an integration, when a user copies a quote from a signed article, a signature for the quote is automatically generated, and an element including both quote and text is put into the clipboard, together with the plain text quote (in practise, this would be a `text/html` element

4 Specifically, we talked with the editor in charge of the platforms and the editor in charge of the digital editorial office at a large media company that produces multiple newspapers for different regional areas, in both paper and digital versions.

and a `text/plain` element). When the user then pastes the quote, a website supporting signatures will use the clipboard element with a signature [W3C21]. One challenge with this approach is that the verification is now performed by the websites, rather than a browser extension, for example. Thus, the user has to trust the website to perform the authentication correctly.

An essential choice is how to divide text into tokens, since any subsequence of the tokens is an allowable quote. Natural choices could be by word, sentence, or paragraph. As a more involved choice, one could also define the tokens at a per-token basis, and simply mark the tokens in the HTML code. A variation of this would be to have a default setting, but to allow the Signer to decide how to split the article into tokens when signing. As a variant, one could also consider using content extraction policies, as in [SBZ01], so the Signer can specify which subsequences of tokens are allowable quotes. A media company might want to disallow quotes of noncontiguous segments, for example, or disallow including only parts of a sentence containing a negative, such as "not", "neither", or "never". Such restrictions could be handled efficiently using regular expressions.

We are implementing a prototype,[5] separated into two parts: a library that can be used by media companies to sign their articles and a browser extension that allows users to quote with signatures and to verify signatures for quotes. The library contains implementations of the relevant algorithms from Section 2.4.1 that each media companies can integrate into their publishing workflow. The browser extension modifies websites such that text (both full articles and quotes) with verified signatures is shown to be signed, and allows the user to make quotes from the signed text that include a signature for the quote. The browser extension also allows the user to get more information from the signature for a quote, e.g., who signed it, when it was signed, an indication of where text was removed, and a link to the original article.

One could further extend the system with different labels, depending on the quality of the source of a quote. For example, many countries have press councils enforcing press ethics, which includes providing correct information, e.g., by researching sufficiently and publishing errata when needed. Hence, it may make sense to mark quotes from articles written by news media certified as following press ethics and rulings of a national press council. One could even go so far as to authenticate only signatures signed by such sources.

To make a difference in the future, media companies and users on social media need to adopt these quotable signatures. To have the best effect, social media platforms should directly support quotable signatures and the required extension should be natively integrated into browsers.

5 To be made available at https://serfurth.dk/research/archive/

2.6 FUTURE WORK

With this paper, we have extended the theory on quotable signatures and presented an application of quotable signatures as a supplementary approach to mitigating the effect of fake news.

Further work on quotable signatures could include using methods similar to the ones employed in [HRS16] and [ABB+22] to remove the requirement that the used hash function be collision-resistant, and thereby remedy a vulnerability against multi-target attacks against hash functions. Additionally, variants of quotable signatures optimized for different types of media should be developed and compared. Our current variant is in some sense optimized for cases where one will often wish to quote something contiguous in one dimension, such as text. If, instead, the goal is to crop an image, one would end up with a "quote" that is contiguous in two dimensions. We have not yet explored how to handle this case effectively. Finally, as discussed in Section 2.5, different policies for dividing text into tokens could be studied.

A natural next step towards using quotable signatures to combat misinformation would be to verify the effectiveness of the proposed method experimentally. In particular, the effects of using quotable signatures for verifying news shared on social media and elsewhere need to be investigated. A suggestion for a first study could be to investigate if the use of quotable signatures improves participants' ability to recall from which news brand a story originated, which was an issue identified in [KFN18]. Additional studies along the lines of [AGB+22], investigating the effects on the quality of the news diet of participants, would also be of interest.

3

DIGITAL SIGNATURES FOR AUTHENTICATING COMPRESSED JPEG IMAGES

Simon Erfurth. Digital signatures for authenticating compressed JPEG images. In *Security-Centric Strategies for Combating Information Disorder - SCID 2024*, page 4. ACM, 2024. DOI: 10 . 1145 / 3660512 . 3665522

ABSTRACT We construct a digital signature scheme for images that allows the image to be compressed without invalidating the signature. More specifically, given a JPEG image signed with our signature scheme, a third party can compress the image using JPEG compression, and, as long as the quantization tables only include powers of two, derive a valid signature for the compressed image, without access to the secret signing key, and without interaction with the signer. Our scheme is constructed using a standard digital signature scheme and a hash function as building blocks. This form of signatures that allow image compression could be useful in mitigating some of the threats posed by generative AI and fake news, without interfering with all uses of generative AI.

Taking inspiration from related signature schemes, we define a notion of unforgeability and prove our construction to be secure. Additionally, we show that our signatures have size 32.5 kb under standard parameter choices. Using image quality assessment metrics, we show that JPEG compression with parameters as specified by our scheme, does not result in perceivably reduced visual fidelity, compared to standard JPEG compression.

3.1 INTRODUCTION

Digital signature schemes are a classical and widely used tool in modern cryptography (the canonical reference is [DH76], and [CMRR23] contains some current standards). For standard digital signature schemes, messages are required to be bit-wise identical to when they were signed, in order to be authenticated. For many applications—for example text—this makes perfect sense, but for others—for example images—it can become a limitation. When an image is shared online, it is very often compressed, typically to save both storage space and communication bandwidth. In general, compression is required to not fundamentally change the content of the image. After compression, it is (typically) not possible to restore the image to be bit-wise identical to the version that was uploaded, so compression invalidates any standard digital signature that

was associated with the image.[1] In order to allow a signature to still be valid after compression, we construct a special digital signature scheme which allows signatures for a JPEG image to be updated during compression of the image, such that the updated signature is valid for the compressed image. Crucially, updating a signature does not require knowledge of the secret key used to generate the signature, yet the updated signature is still signed with the same secret key as the original signature. The only requirement for our signature scheme is that JPEG compression has to be performed using quantization tables containing only powers of two.

Creating multiple signatures for the image is not sufficient, since the signer does not know how the image might be compressed by others. While the signer can control how the image is compressed on the initial website it is posted on, the image might be shared on social media and other websites, where the signer does not control how the image is going to be compressed.

Digital signature schemes for images allowing compression, in one way or another, have been considered before. However, these all suffer from one major drawback or another, such as resulting in compressed images of much lower visual quality [JWL11], only working for the JPEG2000 standard which was never widely adopted [ZSL04], giving only very weak and unclear notions of unforgeability [LC01], or being computationally expensive to compute [DB23]. In contrast, our construction avoids all of these drawbacks, at the cost of being less flexible than some of these schemes.

TECHNICAL OVERVIEW Our signature scheme is constructed to work with JPEG compression, since this is a widely used compression standard, and is used to compress images uploaded to social media. JPEG compression works by using that human vision is more sensitive to some features than it is to others. Concretely, JPEG compression makes use of two specific features: *a*) Humans are more sensitive to changes in luminance than to changes in color, so JPEG compression preserves more information about luminance than color, and *b*) the human vision system is less likely to notice high frequency changes in intensity than low frequency changes in intensity.[2] Thus, if an image contains both an area with high frequency changes in luminance and color shades (like grass or a treetop), and an area with only low frequency changes (like a blue sky), humans will be much more sensitive to small changes in the second area

1 The exemption to this is loss-less compression schemes which allow images to be restored to be bit-wise identical to their uncompressed versions. For these compression schemes standard digital signature schemes will work. However, for many use cases loss-less compression schemes cannot achieve high enough compression ratios, and thus lossy compression schemes are used.

2 Note that here *frequency* does not refer to the wavelength of light, but to the frequency of the intensity of a color/luminance an area of a picture. For example, an area with no change in color/intensity will have very low frequency and an area with where the pixels alternate between being black and white will have very high frequency.

than in the first. Hence, JPEG compression will generally preserve more information about the latter area. This is done by splitting images into into 8×8 pixel blocks, which are then transformed with the discrete cosine transformation (DCT) into a representation using a basis of discrete cosine waves. Now, as the lossy step of JPEG compression, all coefficients are divided by a value from a *quantization table* and rounded. Generally, coefficients corresponding to high frequency changes and/or luminosity are divided by larger values than coefficients corresponding to low frequency changes and/or color. This results in less information being lost due to rounding for features humans are sensitive to, compared to features humans are less sensitive to.

This lossy step is exactly what prevents standard digital signatures from being used for images that will be compressed using JPEG compression. For our signature scheme, this step is also crucial. A key observation is that when the value in the quantization table is a power of two, division and rounding down acts as truncation of the DCT coefficient. The idea is therefore that when an image is compressed with a quantization table containing only powers of two, our signature scheme should still be able to authenticate the bits that have not been truncated. Thus, we create a signature on what is essentially the root of a tree of hashes, where each hash takes as input a combination of hashes from a deeper level of the tree and bits from the DCT coefficients of the image. Now, if an image is compressed with a quantization table containing only powers of two, a signature for the image can be updated to one for the compressed image, by including a subset of the hashes that were made using now truncated bits from the image in the signature. Using that coefficients corresponding to the same DCT basis element in different blocks are always cropped by the same amount, we can create an efficient signature that requires $O(1)$ more work than a regular digital signature, and also result in a signature of constant size (depending only on the choices of parameters, and not on the size of the image). A bonus from the construction of the scheme is that it is fully backwards compatible, in the sense that any JPEG image viewer can display JPEG images compressed with our scheme (even if they might not be able to authenticate the signature for the image). Additionally, our scheme allows repeat compression and updating of signatures, as long as all used quantization tables only contain powers of two, and the signature is kept up to date in each compression.

APPLICATION A use case of our scheme could be for mitigating to consequences of fake news, and in particular to mitigate the threat posed by generative AI for images. The consequences of fake news and misinformation are many, and they pose an increasingly greater threat to democracy and society. Apart from the immediate consequences in the form of uncertainty about whether the content one is looking at is true or not, fake news and misinformation also lead to an increasing level of distrust in the media [OLRW20]. One reason that the threat is increasing is a fundamental change in how news are consumed. Rather than being

consumed directly from news media outlets (such as newspapers, TV, and first-party websites), news is increasingly consumed on social media platforms [NFKN19]. This is an issue, because people tend to be unable to recall which news brand a story originates from when they are exposed to it on social media [KFN18]. Since the news media's image is used as an important heuristic when people evaluate the quality of news [US14], this change allows misinformation to spread. Thus, using digital signatures seems like a natural suggestion; doing so could bind stories shared on social media to the news media that published them, allowing use of this important heuristic. For images specifically, the developments and rise of generative AI over the last few years proves additional cause for concern. With it, everyone can generate (mostly) realistic looking images of anything and everything they can imagine. Considering images' emotional pull and highly persuasive influence [AO09], they are an effective medium for spreading fake news, and hence the sheer volume of (potentially misinforming) images that generative AI can create is highly problematic.

A prevalent method for addressing misinformation on social media involves flagging potential misinformation, employing either manual or automated detection systems. However, studies have consistently demonstrated that flagging problematic content may backfire, exacerbating its adverse impacts [LES+12; SK20]. Attempting to detect and flag all AI generated images is also not an ideal solution. Not only would it hinder legitimate uses of AI generated images, but it could also lead to the generative AI models being trained to not trigger the detection system, as a variation of the widely used generative adversarial network method for training generative AI [GPM+20]. This would result in essentially an arms race between models for detecting and models generating images [Jen24].

Digital signatures can be used to complement the prevalent approach of checking and flagging misinformation. Specifically, if news media sign the images they post, images can be accompanied by signatures signed by news media when they are shared on social media, whether by the news media or by other users. This would provide a guarantee of the provenance of the image (something that is currently missing), helping people evaluate the quality of any news story associated with the image. If only news media meeting a minimum standard of journalistic quality are allowed to sign their pictures, it would also help tell apart quality journalistic content from potential misinformation [BELN23].Specifically, the absence of a digital signature for a picture would be the first red flag, indicating to users that they should be sceptical. For images produced by generative AI, this approach has some clear advantages over a detection-and-flagging approach. With this approach, it is not possible to "just" train the generative AI models to not be detected. Additionally, this approach still allows news media to use AI generated content for their stories. Images shared on social media are compressed when uploaded, so for this approach to work, the signature needs to allow compression of

the image. Therefore, our signature scheme is perfectly suited for this use case.

The idea of mitigating the effects of fake news and misinformation, by using digital signatures to verify the source of images, has been considered by others. One example is the Coalition for Content Provenance and Authenticity (C2PA) [C2P], which involves many companies, including Adobe, the BBC, Google, Microsoft, and Twitter. C2PA focuses on providing a history of an image, i.e., which device was used to capture it, how it has been edited and by whom, etc. However, their approach relies on trusting software used to edit an image to act honestly. Adding support for compression, without needing to compute a new signature, could perhaps increase the versatility of their approach.

STRUCTURE We provide an overview of related work in Section 3.2, and Section 3.3 covers JPEG compression in greater detail. In Section 3.4, we give a generic definition of digital signature schemes for images allowing compression, define an unforgeability notion for such schemes (Section 3.4.2), construct our signature scheme (Section 3.4.3), and show that our scheme is secure with respect to the notion of unforgeability (Section 3.4.4). Finally, we analyze the complexity of our scheme (Section 3.4.5). A visual evaluation of the JPEG compression used by our scheme is done in Section 3.5, where we show that our scheme is almost as good as JPEG compression without any restrictions. To finish up, in Section 3.6, we consider potential optimizations to our scheme, and where further research could go.

OUR CONTRIBUTION We give a generic definition for how a digital signature scheme for images allowing JPEG compression should work, and a natural definition for what it means for such a scheme to be unforgeable. We then describe a specific construction, and prove that (when instantiated using cryptographic secure primitives) it satisfies the notion of unforgeability. Compared to other schemes that allows signed images to be compressed, our scheme trades supporting multiple types of modifications and/or arbitrary JPEG compression for either being more efficient, having a meaningful notion of security, or having higher visual fidelity.

3.2 RELATED WORK

The observation that JPEG compression can act as truncation of the DCT coefficients, and that this could be used to construct a homomorphic digital signature scheme for images, was first made in [JWL11]. The focus of their work is on developing a signature scheme that allows cropping, but as an auxiliary result the authors observe that if every DCT coefficient is truncated by the same amount (in their work *cropped*), their signature scheme also allows some JPEG compression. However, truncating all DCT coefficients by the same amount results in their scheme not making use of the key idea behind JPEG compression; namely that the hu-

man eye is less likely to notice high frequency changes in intensity than low frequency changes. This results in the visual fidelity of images compressed according to their scheme being lower than the visual fidelity of images compressed under standard JPEG compression parameters at similar sizes. Additionally, for JPEG compression to act as truncation, the quantization table need to consists only of powers of two, and hence their approach only allows 8 different quantization tables, leading to their approach being very inflexible, on top of reducing the visual fidelity. We demonstrate these problems in Section 3.5.

HOMOMORPHIC DIGITAL SIGNATURES Homomorphic digital signatures have been studied in a number of different contexts, both for different types of data (images, text, different data structures, etc.) and for different operations that the signature is homomorphic with respect to (cropping, redaction, various set operations, etc.). In the article discussed above [JWL11], their constructed signature can (depending on specific choices) be homomorphic with respect to image cropping, scaling, or (very restricted) JPEG compression. Other homomorphic signature schemes for images have been considered, in particular for the JPEG2000 image standard. For example, [ZSL04] gives an overview of signatures for JPEG2000 images that are homomorphic with respect to extraction of various representations from single code streams. However, the JPEG2000 image standard was never widely adopted, and the only web browser supporting JPEG2000 is Safari [Ado].

Much more generally, [ABC+15] provides a generic definition and construction of homomorphic signatures, which allows deriving a signature on m' from a signature on m, as long as $P(m,m') = 1$ for a (univariate and closed) predicate P. Their definition encompasses many examples of individual well-studied homomorphic signatures, including arithmetic, quotable, redactable, and transitive signatures [KFM04; ZKMH07; BELN23; SBZ01; JMSW02; MR02; BN02], but the generic scheme is generally less efficient than more specialized schemes.

PERCEPTUAL HASHING Another approach that in principle gives rise to digital signatures allowing image compression, is the use of perceptual hashing [MH80] in place of the cryptographic hash functions typically used by digital signature schemes. Conceptually, a perceptual hash function is a hash function where the hash of an image only changes if the images is *perceivably* different enough. Thus, JPEG compression (to some extent) should not change the perceptual hash of an image. Combined with a standard digital signature scheme, a perceptual hash function should therefore create a digital signature scheme allowing JPEG compression. The issue with this approach is that all perceptual hashes known to the author have an overlap between having false positives and having false negatives. That is, all schemes will either accept (randomly) manipulated images as being authentic with a non-negligible probability, or they will reject authentic images as being manipulated with a non-negligible prob-

ability [DHC20]. Thus, perceptual hashing cannot reasonably be used in place of a cryptographic hash in a digital signature scheme.

Considering JPEG compression specifically, Lin and Chang [LC01] created a perceptual hash function with this in mind. They find relationships between the 8×8 pixel blocks in an image that are somewhat invariant under JPEG compression. However, even this approach allows a manipulated image to be accepted with non-negligible probability. Specifically, the authors perform a practical experiment where they change a random block, and, using various parameters, find that the probability of a manipulated image being accepted is between 0.04 and 0.00001 when the image is not compressed, between 0.09 and 0.0002 with a quality factor of 50, and between 0.2 and 0.02 with a quality factor of 20. That is despite these attacks assuming that the manipulator had no knowledge of which blocks were being compared, and made a non-targeted attack by editing one chosen at random. An attacker with knowledge of which blocks are being compared could potentially make a targeted attack with even higher success chance. In contrast, it follows from the correctness of our scheme, that we have no false negatives, and from unforgeability that there is only a negligible probability of false positives.

CRYPTOGRAPHIC APPROACHES TO MITIGATING MISINFORMATION THROUGH IMAGES Using digital signatures to prevent misinformation through images has also been considered in [AJAZ22]. In this article, the authors suggest using standard digital signatures directly on images, and focus more on the technical considerations for how this could be implemented. One obstacle to using their approach is that in practice, images are almost always compressed when uploaded online (for example to social media) and all standard digital signature schemes require the image to be bit-for-bit identical to the signed image. In order to fix this shortcoming, their work could be changed to instead use our digital signature scheme, in which case it considers the technical details of implementing digital signatures allowing compression.

In [SIM+22], the authors discuss on a more general level how cryptographically proving provenance can be a proactive partial solution to mitigating misinformation. Based on literature from human-centered computing and usable security, journalism, and cryptography, they consider both advantages and challenges of such a system, and find properties a system should have.

A different suggestion for using cryptography to prevent misinformation through images is made in [DB23]. They suggest using succinct non-interactive zero-knowledge proofs to verify the metadata of an image, and that the image has only been modified in some claimed ways. More concretely, their suggested solution is to use a camera that adds metadata to an image when it is taken, and signs the image and metadata (such a camera was recently released by Leica in collaboration with the C2PA [Lyo23]). When an image is later edited, the editor also generates a zero knowledge proof of the following statement: *"The prover (i) knows*

an unedited photo that is properly signed by a C2PA camera, (ii) the metadata on the unedited signed photo is the same as the one attached to the public photo, and (iii) the public photo in the news article is the result of applying the claimed edits to the unedited photo." Before an image is displayed, the zero-knowledge proof is verified, and the image is then displayed together with its accompanying metadata. While their construction allows arbitrary compression, and many other operations, their solution does not appear to be efficient enough to use for images shared on social media. Generating a proof takes minutes, even on a modern system, and considering how many images are uploaded to social media, it would not be feasible to generate proofs for all of them. Prior to [DB23], it was suggested in [NT16] to use zero-knowledge proofs to authenticate that only permissible transformations has been made to an image. However, the proving time of their implementation is even longer (around 5 minutes to generate proofs for 128×128 pixel images). In contrast, our construction requires at most 1025 hash function evaluations and one key generation, signing, or verification of a standard signature scheme.

3.3 JPEG COMPRESSION

In order to construct our signature scheme, we need a general understanding of how JPEG compression works. For each step, more information can be found in [Wal91], and in the official JPEG standard [Int92].

As we mentioned in the introduction, the key observation behind JPEG compression is that humans are much better at noticing some types of details than others. In particular, humans are less likely to notice high frequency changes in intensity of both color shades and luminance, than low frequency changes. Similarly, humans are less likely to notice changes in color compared to changes in luminance. JPEG compression uses this to perform lossy compression without sacrificing too much perceived image quality, by preserving more information about the coefficients for low frequency changes and about luminance, and less information about high frequency changes and about color.

The first step of JPEG compression is to convert the image to the YCbCr color-space,[3] which, later in the process, allows preserving more information about luminance than about color. As an optional second step, the color channels can be down-sampled either just along one axis, or along both axes, meaning that the resolution of one or two dimensions is halved: a mean value of two (or four) pixels is found, and used for two (or four) pixels. Steps three and four are applied to each 8×8 block of the image separately (with appropriate padding when the image dimensions are not multiples of 8). As the third step, the discrete cosine

3 Meaning that instead of representing the image using red, green, and blue channels (RGB), it is represented as one luminance channel (Y) and two color channels (Cb and Cr). Mathematically, this is a lossless transformation, but in practice there will of course be some losses due to rounding errors. For simplicity, we assume that before this step, all images are 8 bit RGB images. However, all constructions are easily changed to work on 10-bit images.

transformation is applied to the 64 values to transform them into 64 DCT coefficients. Roughly, this can be thought of as changing each pixel from representing the intensity of that specific pixel into representing the intensity of a particular (discrete) cosine function over the entire 8×8 pixels block; see Figure 14a. This step allows preserving more information for the low frequency cosine waves that represents low frequency changes in the image. The fourth step is referred to as quantization and is the only lossy part of JPEG compression. In this step, each entry of the 8×8 pixel block is divided by an entry from the *quantization table* and rounded. The quantization table is an 8×8 table, consisting of values in the range 1 to 256. Generally, entries representing low frequency cosine waves in the block are divided by smaller values from the quantization table, and entries representing high frequency changes are divided by larger values. Hence, less information is lost for low frequency cosine waves due to rounding, i.e., when the process is reversed by multiplying with the entries from the quantization table, the coefficients for low frequency cosine waves generally end up closer to their original value than coefficients for high frequency cosine waves. By choosing different values for the quantization table, it is possible to control the trade-off between how much the file size is reduced and how much the quality of the image is reduced. Finally, the image is encoded using a lossless entropy encoding, usually Huffman encoding. A number of tricks are applied as preprocessing in this step, but for brevity (and since they are not directly relevant to this work), we will not discuss them, but refer instead to [Int92, Section 4.3], and the related standards. To display an image, each of these steps are reversed in the opposite order.

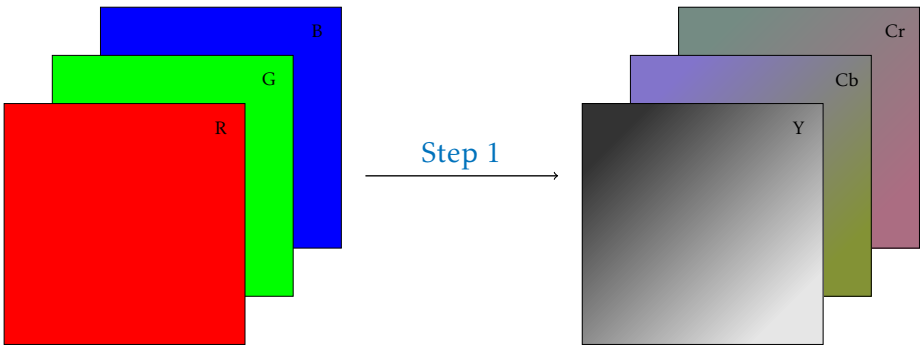
To summarize, JPEG compression consists of the following steps, which we have also illustrated in Figures 13a, 13b, 14a and 14b.

1. Convert the image from RGB to the YCbCr color-space.
2. Optionally down-sample the color channels (Cb and Cr).
3. For each 8×8 pixels block in each channel:
   a. Apply the discrete cosine transformation to the block.
   b. Quantizise the block, using the quantization table.
4. Encode the image using a lossless entropy encoder.

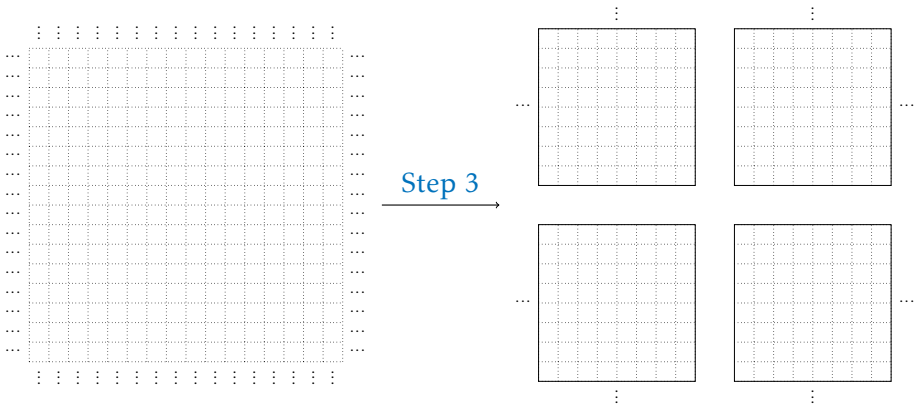
3.3.1 *DCT Transformation*

Step 3a, illustrated in Figure 14a, is a change of basis from using the standard basis for 8×8 matrices to the DCT basis. That is, instead of using the basis $\{e_{i,j}\}_{i,j \in \{0,\dots,7\}}$, where

$$(e_{i,j})_{p,q} = \begin{cases} 1 & \text{if } i = p \text{ and } j = q \\ 0 & \text{otherwise,} \end{cases} \tag{9}$$

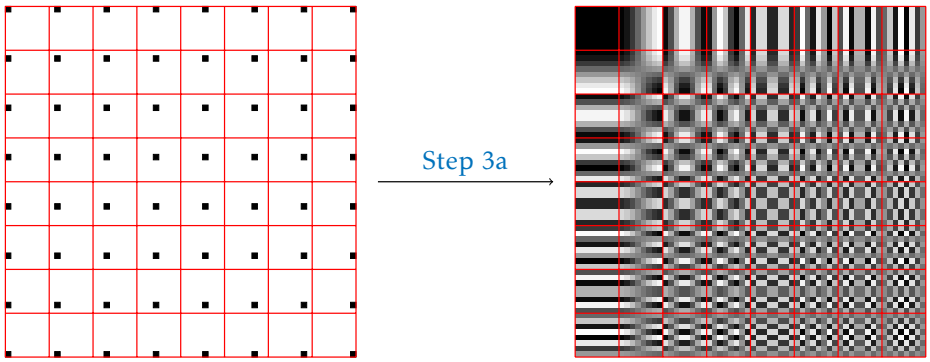


(a) Transformation from RGB color space to YCbCr color space.



(b) Splitting into 8×8 pixel blocks.

Figure 13: Select steps of JPEG compression.



(a) Changing to DCT representation, e.g., instead of each pixel representing its own value, it now represents one of the DCT basis functions. As illustrated in the figure, this can also be thought of as a change of basis from the standard basis to the DCT basis.

815	-15	-10	-2	-2	-3	3	3
-3	2	-9	-4	4	-2	-2	3
0	1	-2	-4	-2	-2	2	-1
-3	-1	-3	3	1	0	3	1
0	-1	2	3	-1	1	-1	1
-5	6	1	-1	4	-1	-1	-3
2	1	2	0	0	7	3	-1
0	1	3	-3	-2	3	1	-5

Step 3b

51	-1	-1	0	0	0	0	0
0	0	-1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

(b) Quantizise each block using the quantization table. Observe how information is preserved for the values corresponding to low-frequency DCT basis elements in the upper left corner. Values are the luminance values of a random block from the Lenna test image, quantized using the quantization table generated with $Q = 50$, as discussed in Section 3.3.2.

Figure 14: Select steps of JPEG compression.

we now use the basis $\{B_{i,j}\}_{i,j\in\{0,\dots,7\}}$, where

$$(B_{i,j})_{p,q} = \cos\left(\frac{\pi(2p+1)i}{2\cdot 8}\right)\cdot\cos\left(\frac{\pi(2q+1)j}{2\cdot 8}\right) \tag{10}$$

for $p,q \in \{0,\dots,7\}$. The resulting discrete cosine functions are illustrated on the right side of Figure 14a. We stress that what is stored in the 8×8 matrices are the *coefficients*, $a_{i,j}$ and $a'_{i,j}$, of the basis elements, i.e., $a_{i,j}$'s and $a'_{i,j}$'s such that the 8×8 block can be expressed as

$$\sum_{i,j} a_{i,j}e_{i,j} = \sum_{i,j} a'_{i,j}B_{i,j}. \tag{11}$$

Since both the standard basis and the DCT basis are spanning, this is a lossless operation (up to rounding). As mentioned, the purpose of this step is to allow the next step to preserve more information for low frequency changes, which will generally be represented by the basis elements where i and j are small; see Equation (10) and the right side of Figure 14a.

3.3.2 *Quantization*

In Step 3b of the JPEG compression, each value in the 8×8 block of DCT coefficients is quantized by dividing it by a value from a quantization table and rounding the result. Despite the standard [Int92] specifying that the result should be rounded towards the nearest integer, this appears to not be the case in practise. As an example, the widely used implementation from The Independent JPEG Group (IJG) [The22] rounds towards 0. When displaying a compressed image, the value is multiplied with the value from the quantization table again. Hence, information can be lost in the step. For example, different values may end up being mapped to the same, see the first row of Figure 14b.

As mentioned earlier, one of the tricks that JPEG compression uses is that the human eye is much more sensitive to low frequency changes. Hence, it will generally be the case that the values in the quantization table grow as one goes from the upper left corner to the lower right. Additionally, since the human eye is more sensitive to luminance than color, different tables are used for the luminance channel, Y, and for the color channels, Cb and Cr.

However, there are principally no requirements for quantization tables (apart from them being 8×8 tables with values between 1 and 256), and indeed, this is one of the main points where various programs implementing JPEG compression can differ.[4] However, one standard approach is to let the *quality factor* p be an integer with $1 \le p \le 100$, fixing the table for quality factor 50 to be Q_{50}, and then deriving the other tables from Q_{50}. For example, in the IJG implementation [The22],

4 Section 3 of [TB14] gives an overview of quantization tables for different purposes.

$$Q_{50} = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix} \tag{12}$$

and for $1 \leq p \leq 100$ we have

$$Q_p = \begin{cases} \frac{50}{p} Q_{50} & \text{if } p < 50 \\ \frac{200 - 2p}{100} Q_{50} & \text{otherwise.} \end{cases} \tag{13}$$

Finally, the entries in Q_p are rounded, and it is ensured that every entry is at least 1 and at most 255. This definition means that quantization with Q_{100} does not destroy information (Q_{100} is all 1's), and as the quality factor decreases, the values in Q_p increase, so that more detail is lost, but the compression also results in a smaller file.

3.4 SIGNATURE CONSTRUCTION

Sections 3.4.1 and 3.4.2 contains a generic definition of what a digital signature scheme for images allowing compression is, and what it means for it to be unforgeable. From Section 3.4.3 and onward, we construct such a signature scheme, show that it is unforgeable, and analyse its performance.

3.4.1 *Generic Definition*

In general, a digital signature scheme for images allowing compression consists of four efficient algorithms, KeyGen, Sign, Compress, and Verify. These are essentially the three standard digital signature algorithms for key generation, signing, and verification, with an added compression algorithm. The compression algorithm allows a signature for an image to be updated to a signature for a compressed version of the image, given both the original image, the signature for the original image, and the compressed image. For simplicity of further definitions, we make the abstraction that the compression algorithm also performs the compression. We use λ to denote the security parameter. Summarizing, we require that the four algorithms acts as follows.

- $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ takes as input the security parameter 1^λ, and outputs a key pair.

- $\sigma \leftarrow \mathsf{Sign}_{\mathsf{sk}}(i)$ takes as input a secret key sk and an image i. Outputs a signature for i that allows compression.
- $(i', \sigma') \leftarrow \mathsf{Compress}(i, \sigma, P)$ takes as input an image i, a signature σ for i, and additional parameters P specifying how compression should be done. In the case where the compression is JPEG compression, P is the quantization tables. Outputs the compressed image i' and a signature σ' for i', that is still signed with the private key that σ was signed with. Note that i and σ could have been obtained from an earlier compression.
- $\top / \bot \leftarrow \mathsf{Verify}_{\mathsf{pk}}(i, \sigma)$ takes as input a public key pk, an image i, and a signature σ for i. Outputs $\top$ if σ' is a valid signature for i with respect to pk, and $\bot$ otherwise.

A standard notion of correctness should be satisfied, meaning that a genuine signature should always be accepted.

3.4.2 *Security Notion*

Our notion of security takes inspiration from other digital signature scheme variants allowing some form of modification, e.g., redactable signatures [SBZ01; JMSW02], quotable signatures [BELN23], the image signatures from [JWL11], and generic P-homomorphic signatures [ABC+15]. Essentially, these notions of security say that the scheme is unforgeable if no adversary can produce a signature for a message that is not either a message the signing oracle has provided a signature for, or the result of performing an "allowed" operation on a message the signing oracle has provided a signature for. In our case, this means that the image the adversary outputs cannot be obtained by performing allowed compression on any of the images queried to the signing oracle. To make the definition general, we define an "allowed compression" to be any compression that can be done by Compress under valid input. For our scheme specifically, the definition says that no adversary can output a signature for an image that is not either an image the adversary has queried the signing oracle for, or the result of compressing one of these images using quantization tables that consists of only powers of two. We formally define this in Definition 3.1.

> DEFINITION 3.1 *Unforgeability.*
> For a signature scheme
>
> $$\mathsf{CS} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Compress}, \mathsf{Verify}) \tag{14}$$
>
> allowing image compression, we define the compression span of Compress on an image I with valid signature σ to be
>
> $$\mathsf{CSpan}(I, \sigma) := \{I' \colon (I', \sigma') \leftarrow \mathsf{Compress}(I, \sigma, P)\}, \tag{15}$$

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
$(i^*, s^*) \leftarrow \mathcal{A}^{\mathsf{Sign}_{\mathsf{sk}}(\cdot)}(\mathsf{pk})$
// denote the queries that $\mathcal{A}$ make to the signing oracle by $i_1, i_2, \ldots, i_Q$,
// and the answers by $\sigma_1, \sigma_2, \ldots, \sigma_Q$.
if $(\mathsf{Verify}_{\mathsf{pk}}(i^*, s^*) = \top) \wedge (\forall k \in \{1, 2, \ldots, Q\} \colon i^* \notin \mathsf{CSpan}(i_k, \sigma_k))$
return 1

Figure 15: The Unforgeability experiment. We wqrite $\mathcal{A}^{\mathsf{Sign}_{\mathsf{sk}}(\cdot)}$ to indicate that $\mathcal{A}$ is given access to an orcale that simply signs images under sk using Sign.

where P is valid extra parameters to Compress. That is, $\mathsf{CSpan}(I)$ is the set of all images that I can be compressed to by Compress. The signature scheme CS is said to be *existentially unforgeable*, if for every probabilistic polynomial time adversary $\mathcal{A}$, the probability of the experiment in Figure 15 returning 1 is negligible.

3.4.3 *Our construction*

Conceptually, our idea builds on the observation that if all the entries in the quantization table are powers of two, then we can consider the quantization step as being *truncation* of the least important information. This allows us to construct a signature where one can provide some (small) piece of information that allows a signature for an image to be verified, even if the image has been compressed.

To illustrate the idea, we consider an example where we have just one 8 bit value, $b = b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0$, which we wish to sign in a way that allows us to truncate a number of (least significant) bits. This can be done by constructing a *chain of hashes* as follows. The first node in the chain is the hash of the least significant bit, $H(b_0)$. Any other node is the hash of the concatenation of the previous node and the next least significant bit, i.e. the second node will be $H(H(b_0) \| b_1)$. Finally, one signs the last node in the chain (the *end node*) using a standard digital signature scheme. We illustrate this in Figure 16.

Now it is possible to truncate out some of the least significant bits of b, and, if one instead provides the node of the chain of hashes corresponding to the most significant of the truncated bits, the signature for b can still be authenticated, since h_7 can still be computed. This is done by calculating the chain of hashes, starting from the node of the least significant bit that was not truncated, which can be calculated using the provided value. In Section 3.4.4 we show that this still binds the non-truncated bits, in the sense that these bits cannot be changed without invalidating the signature. We also argue that for our use case, this signature provides a meaningful notion of security. Of course, for this example, it would have been much more space efficient to send the truncated bits instead of a hash value. However, as we see in Section 3.4.5, this is

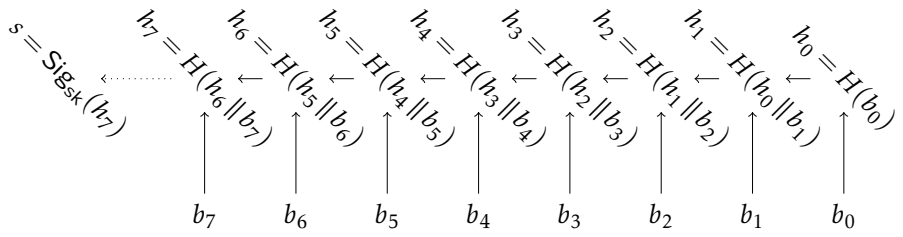


Figure 16: The chain of hashes and signature for one byte $b = b_7b_6b_5b_4b_3b_2b_1b_0$.

not the case when we consider (larger) images, rather than just single bytes.

Going back to full images, we recall that JPEG compression works on 8×8 pixel blocks, where every block is handled the same way. In particular, the coefficient for a specific DCT basis element will be truncated by the same number of bits in every block. Thus, we only need one chain of hashes for each basis element, regardless of the size of the image. For any basis element $B_{i,j}$, the first node in the chain of hashes is the hash of the concatenation of the least significant bits of all bytes at location i, j in each of the 8×8 blocks, in an arbitrary, but fixed, order. For all other nodes in the chain of hashes, the node is the hash of the concatenation of the previous node and the next least significant bits of all bytes at location i, j. In Figure 17, we illustrate how the chain of hashes is calculated for the entries corresponding to the first DCT basis element, $B_{0,0}$.

For each of the two quantization tables (one for luminance and one for color), we calculate the chain of hashes for each of the $8 \cdot 8 = 64$ DCT basis elements, obtaining 128 end nodes, one for each basis element in each channel. The end nodes are then hashed together, in order to get one final hash, h_{root}, which is signed using a standard digital signature scheme. This final process is illustrated on Figure 18.

Now any party can compresses the image while still allowing the signature to be verified, by performing the compression using a quantization table with only powers of two in it, and providing the relevant nodes from the 128 chains of hashes. To be specific, if entry (i, j) in the quantization table is 2^k, the compressing party adds node h_{k-1} from the chain of hashes corresponding to $B_{i,j}$ to the signature (if $k = 0$ no compression is done, and no node is added). We denote the added nodes as *the truncated hashes*.

To summarize, our digital signature scheme allowing some JPEG compression works as follows, where H is a cryptographic hash function and $\mathsf{DS} = (\mathsf{KeyGen}^{\mathsf{DS}}, \mathsf{Sign}^{\mathsf{DS}}, \mathsf{Verify}^{\mathsf{DS}})$ is a standard digital signature scheme.

- $\mathsf{KeyGen}(1^\lambda)$: Identical to $\mathsf{KeyGen}^{\mathsf{DS}}(1^\lambda)$.
- $\mathsf{Sign}_{\mathsf{sk}}(i)$: Compute the 128 chains of hashes, as described above, compute the hash of the end nodes to obtain h_{root}, and sign this using $\mathsf{Sign}^{\mathsf{DS}}$.

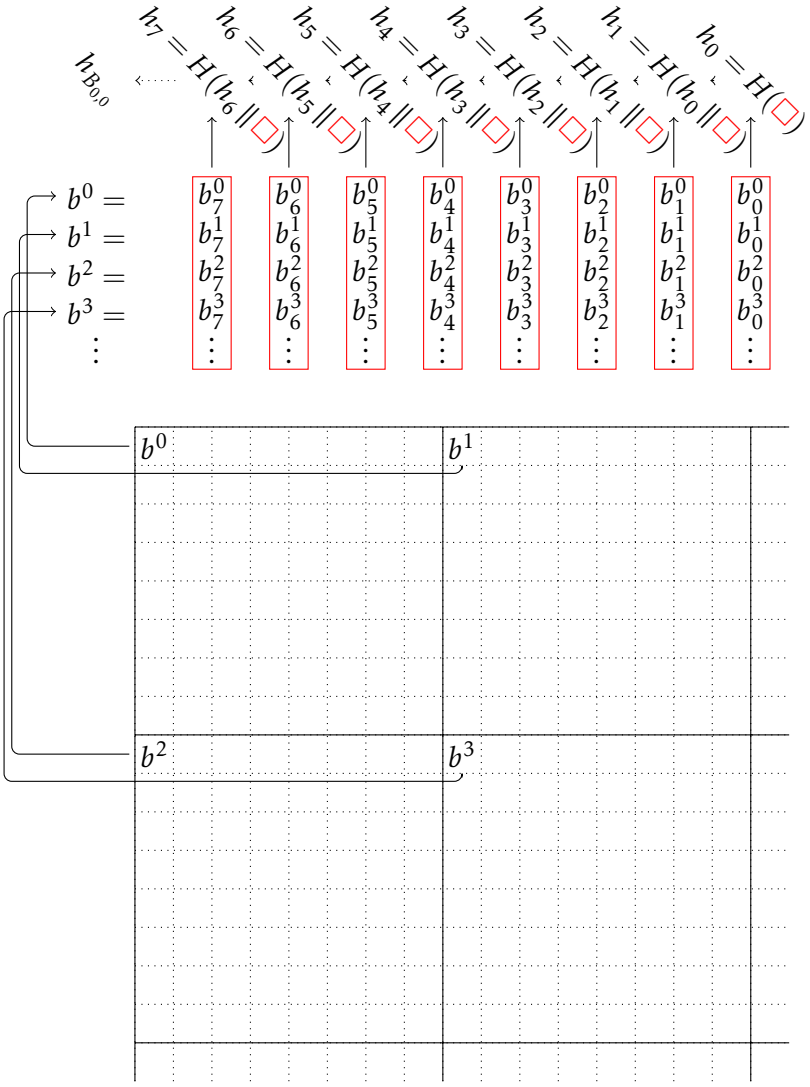


Figure 17: Example illustrating construction of one of the chains of hashes.

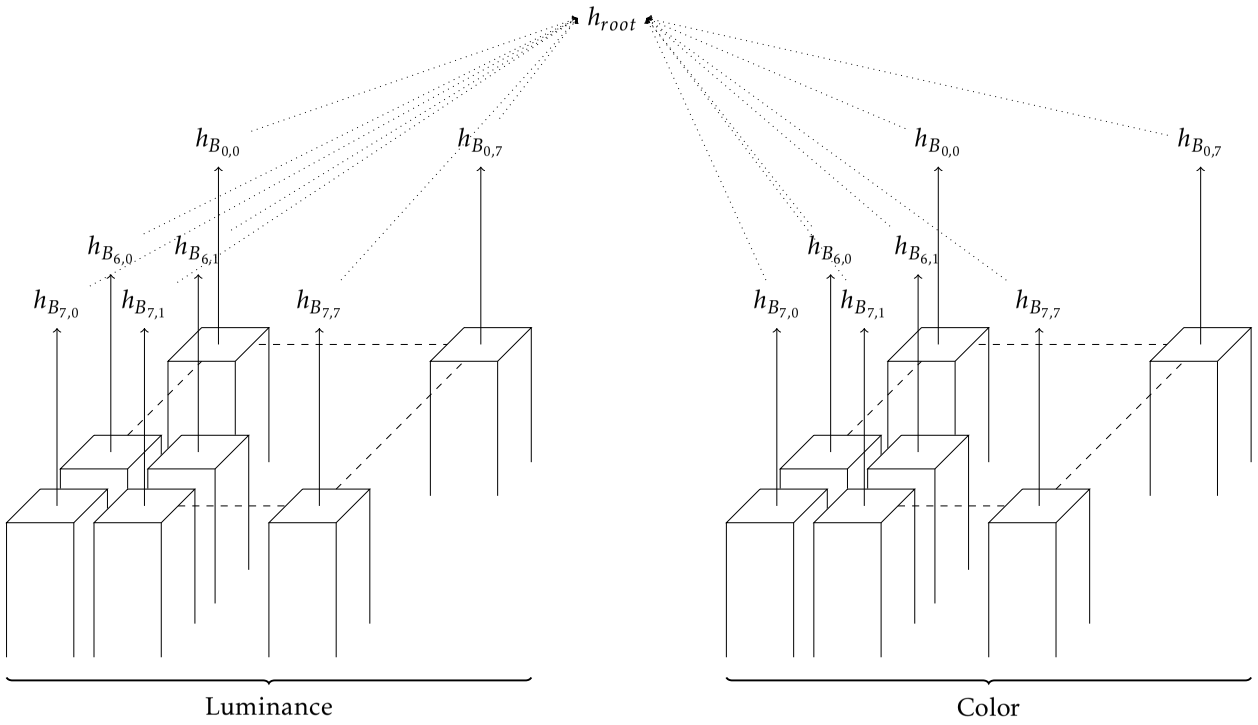


Figure 18: The 128 end nodes (64 from the to the luminance channel and 64 from the color channels) are used to calculate h_{root}. Then h_{root} is signed using the standard digital signature scheme, obtaining the digital signature for the uncompressed image.

- $\mathsf{Compress}(i, \sigma, P)$: Given an image, a signature for the image, and quantization tables containing only powers of two, first perform standard JPEG compression, always rounding down. Then compute the chains of hashes from the original image, and extract the 128 truncated hashes. Together with the signature for the uncompressed image, the truncated hashes form the signature for the compressed image. Output the compressed image and the signature for the compressed image. Observe that a compressed image can be further compressed (using a quantization table with at least as large powers of two) and the signature updated with new truncated hashes to be valid for the further compressed image.

- $\mathsf{Verify}_{\mathsf{pk}}(i, \sigma)$: Use the image, and if it is compressed the truncated hashes, to find h_{root}, and verify with respect to the original signature using $\mathsf{Verify}^{\mathsf{DS}}$.

Note that this matches the description of digital signature schemes allowing compression, outlined at the end of Section 3.4.1.

3.4.4 *Security Analysis*

Theorem 3.2 shows that if the primitives used in the scheme constructed in Section 3.4.3 are secure, the scheme is secure in the sense of Definition 3.1. We show this by arguing that an adversary for our scheme can be used to construct an adversary for at least one of the primitives.

> THEOREM 3.2
> Under the assumption that
>
> - H comes from a family of cryptographic secure hash functions,
> - $\mathsf{DS} = (\mathsf{KeyGen}^{\mathsf{DS}}, \mathsf{Sign}^{\mathsf{DS}}, \mathsf{Verify}^{\mathsf{DS}})$ is an existentially unforgeable standard signature scheme,
>
> $\mathsf{CS} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Compress}, \mathsf{Verify})$ constructed as described above, is an existentially unforgeable signature scheme allowing image compression.

Proof. Assume that $\mathcal{A}$ is a probabilistic polynomial time adversary against the unforgeability of CS, as defined in Definition 3.1. We show that the probability of $\mathcal{A}$ being successful is negligible. Let (i^*, s^*) be the output $\mathcal{A}$, with $s^* = (\mathsf{Sign}^{\mathsf{DS}}_{\mathsf{sk}}(h_{root}), \{h_k^{i,j,c}\})$, i.e., s^* consists of the standard signature for h_{root} of i^*, and the (possibly empty) set of relevant nodes from the chains of hashes, indexed by i, j, c, where i, j specifies a DCT basis element, and c specifies if it is from the luminance or the color quantization table.

Consider first the case where h_{root} of i^* (which can be found using i^* and $\{h_k^{i,j,c}\}$) is different from h_{root} of each of the Q images $i_1, \ldots, i_Q$ that were $\mathcal{A}$'s queries to the signing oracle. In this case, (h_{root}, s^*) is a forgery

against DS, and since DS is assumed existentially unforgeable, this can happen with at most negligible probability ϵ_{DS}.

For the other case, let $\hat{i}$ denote the image queried to the signing oracle such that h_{root} of $\hat{i}$ is the same as h_{root} of i^*. In the following, we will use $\hat{h}$ to indicate that a hash is generated from $\hat{i}$, and just h for values generated from i^*. With this notation, we are considering the case where $\hat{h}_{root} = h_{root}$. We now compare each of the 128 hash values (end nodes) that were used to generate $\hat{h}_{root}$ and h_{root}. That is, for $i,j \in \{0,\dots,7\}$ and $c \in \{\mathrm{Y},\mathrm{Cb/Cr}\}$, we compare $\hat{h}^{i,j,c}_{B_{i,j}}$ and $h^{i,j,c}_{B_{i,j}}$. If $\hat{h}^{i,j,c}_{B_{i,j}}$ and $h^{i,j,c}_{B_{i,j}}$ differ in at least one location i,j,c, we have found a collision for H.

If $\hat{h}^{i,j,c}_{B_{i,j}}$ and $h^{i,j,c}_{B_{i,j}}$ are the same at every location, we now go one step further down, and look at each chain of hashes. That is, for each location i,j,c, we consider now $\hat{h}_k$ and h_k for $k \in \{0,\dots,7\}$ (as illustrated on Figure 17). There must be at least one location i,j,c,k where the bits from the image used to calculate $\hat{h}_k$ are different from the bits from i^* used to generate h_k. To see this, observe that if not, one could obtain i^* from $\hat{i}$ by compressing $\hat{i}$ with suitable values in the quantization tables (the values being the ones that truncate the chains of hashes for $\hat{i}$ to the length of the chains of hashes for i^*, possibly 1 if no compression was performed). If $\hat{h}_k = h_k$, this is clearly a collision for H. On the other hand, if $\hat{h}_k \neq h_k$, consider instead $\hat{h}_{k+1}$ and h_{k+1}. If $\hat{h}_{k+1} = h_{k+1}$, we have instead found a collision here, since

$$H(\hat{h}_k \,\|\, \cdots) = \hat{h}_{k+1} = h_{k+1} = H(h_k \,\|\, \cdots), \tag{16}$$

where $\cdots$ indicates that the relevant bits from the images are inserted. Alternatively, we have $\hat{h}_{k+1} \neq h_{k+1}$, in which case and continue on to consider the next nodes in the chain. Since the end nodes of the chains are the same, i.e., $\hat{h}_{B_{i,j}} = h_{B_{i,j}}$, we are guaranteed that the nodes will eventually be the same, and hence we are guaranteed that we find a collision for H. In all cases where we did not find a signature forgery, we have instead found a collision for H. Since H is assumed to come from a family of cryptographic secure hash functions, this happens with at most negligible probability ϵ_H.

It follows that the probability of $\mathcal{A}$ being successful is at most $\epsilon_{\mathsf{DS}} + \epsilon_H$, which is negligible. □

3.4.5 *Performance Analysis*

From the construction of our signature scheme, it is immediate that the signature size is bounded by a constant, depending only on the size of the output of the hash function used and the size of the underlying standard digital signature scheme, see Corollary 3.3. For a hash function with a 256 bit output (for example, the widely used SHA3-256), the scheme has a signature size of at most $128 \cdot 256 = 32{,}768$ bits or 4 kB on top of the standard digital signature. Section 3.6 includes suggestions for making the signature smaller.

Table 4: Upper bounds on the performance of our construction of a signature scheme allowing image compression, assuming it is constructed with a hash function with output size $|H|$ and a standard digital signature scheme DS with signature size $|S|$.

	Computation Time	Signature Size
Key generation	Same as $\mathsf{KeyGen}^{\mathsf{DS}}$	—
Signing	1025 hashes and time of $\mathsf{Sign}^{\mathsf{DS}}$	$\|S\|$
Compression	1025 hashes	$128\|H\| + \|S\|$
Verification	1025 hashes and time of $\mathsf{Verify}^{\mathsf{DS}}$	—

Corollary 3.3
If H is a hash function with output size $|H|$ and DS is a standard digital signature scheme with signature size $|S|$, the scheme CS constructed as described above, has signature size $|S|$ for uncompressed images and signature size

$$128 \cdot |H| + |S| \tag{17}$$

for compressed images.

Proof. For an uncompressed image, the signature for the image consists of the standard digital signature for h_{root}. For a compressed image, the signature consists of the standard digital signature for h_{root} and (at most) one node from each of the 128 chains of hashes, see Figure 18. □

In terms of computation requirements, it follows from the construction of the signature scheme that key generation, signing, and verification require one key generation, signing, or verification from the standard digital signature scheme. Additionally, for signing, compression, and verification, it is necessary to compute h_{root}. Doing so requires computing each of the 128 chains of hashes, each of which requires computing (up to) 8 hashes, plus a final hash to obtain h_{root}. When the image is compressed, verification requires computing fewer than 8 hashes per chain. In total, computing h_{root} therefore requires up to

$$128 \cdot 8 + 1 = 1025 \tag{18}$$

hash function evaluations. Table 4 summarizes the performance of our scheme, in terms of the size of the signature before and after compression, and computation required by each of the four algorithms.

To provide some context, Figure 19 shows the ratio between the size of an image signed with our signature and the size of an image that is not signed, and also the ratio between the size of an image signed with our signature and directly signing it with a standard digital signature (and

hence having to provide the entire image, all the time). For this comparison we consider an image size of 2 megabytes, a hash function with 256 bit output and a standard digital signature scheme with signature size 512 bits. These parameters correspond to using our scheme with SHA3-256 and EdDSA using the Ed25519 curve, both of which are currently thought to be secure, and widely used [Dwo15; CMRR23]. Choosing these parameters result in a signature size of $128 \cdot 256 + 512 = 33,280$ bits, or 32.5 kb, for our scheme. Figure 19 illustrates that, for these parameters, using our signature scheme adds just under 4% overhead when images are compressed down to 5% of their original size. Replacing EdDSA with a post-quantum signature scheme (and thus making our scheme post-quantum secure), increases the overhead of our scheme, but does not fundamentally change the figure. For example, using the post-quantum signature scheme Dilithium in the form suggested by NIST [Nat24a], changes the signature size to be between 2420 and 4595 bytes, rather than 512 bits. Even in the 4595 bytes case, the overhead is only just over 8%, when images are compressed down to 5% of their original size. In this case, our scheme has signature size $128 \cdot 256 + 36,760 = 69,528$ bits, or just under 8.5 kB. Finally, Figure 19 also illustrates that, rather obviously, our scheme performs drastically better than just signing the hash of the image directly, since in this case the entire image has to be provided to verify the signature, and thus no compression can be done.

3.5 VISUAL EVALUATION

An essential question to ask about a construction that modifies how JPEG compression is performed is how it impacts the image quality. In order to evaluate this, we used the IJG implementation [The22] with both the standard quantization tables and quantization tables that contained only powers of two (as in our construction and [JWL11]). For any fixed image, we chose the quantization tables with powers of two to be the tables giving the size closest to the size obtained with standard quantization tables. For our construction, we found the tables in the following way. First, a function r rounding a value v to either the closest smaller or the closest larger power of two is defined. Rather than just rounding v to the closest value, the function takes an additional parameter $q \in [0,1]$, which defines where the cutoff between rounding down and rounding up is. The function is defined as:

$$r(v,q) = \begin{cases} 2^{\lceil \log(v) \rceil} & \text{if } v > (1+q) \cdot 2^{\lfloor \log v \rfloor} \\ 2^{\lfloor \log v \rfloor} & \text{otherwise.} \end{cases} \tag{19}$$

This allows us to tweak q in order to get the size of the image compressed with the generated quantization table as close to the size of the image compressed with the standard quantization tables as possible. For a given quality factor p (and hence a pair of standard quantization tables), we

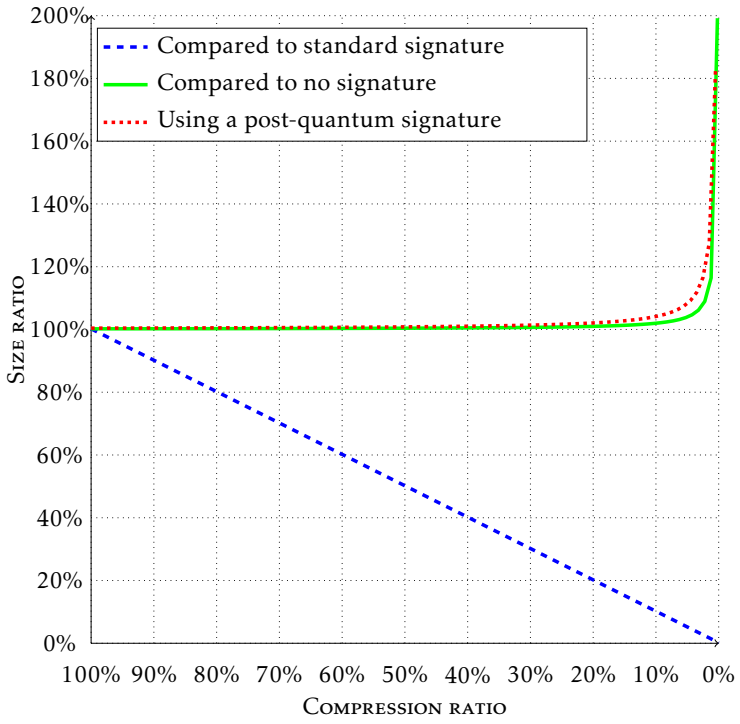


Figure 19: Relative size of an image and signature from our scheme (image size + provided nodes + standard signature) compared to not signing the image at all (image size) in dashed blue and compared to using a standard digital signature scheme (uncompressed image size + standard signature) in solid green. Additionally, relative size compared to not using a signature, when our scheme uses a post-quantum signature in dotted red. For this comparison, we consider an image with an uncompressed size of 2 megabytes, a hash function with output size 256 bits, a standard digital signature scheme with signature size 512 bits, and a post-quantum digital signature scheme with signature size 36760 bits.

perform a binary search to find the value of q that gives the closest compressed image size. As an example, using the image in Figure 20a and quality factor $p = 50$, we obtain the following luminance quantization table from the luminance quantization table in Equation (12):

$$Q = \begin{bmatrix} 16 & 8 & 8 & 16 & 16 & 32 & 64 & 64 \\ 8 & 8 & 16 & 16 & 32 & 64 & 64 & 64 \\ 16 & 16 & 16 & 16 & 32 & 64 & 64 & 64 \\ 16 & 16 & 16 & 32 & 64 & 64 & 64 & 64 \\ 16 & 16 & 32 & 64 & 64 & 128 & 128 & 64 \\ 16 & 32 & 64 & 64 & 64 & 128 & 128 & 64 \\ 64 & 64 & 64 & 64 & 128 & 128 & 128 & 128 \\ 64 & 64 & 64 & 128 & 128 & 128 & 128 & 128 \end{bmatrix}. \tag{20}$$

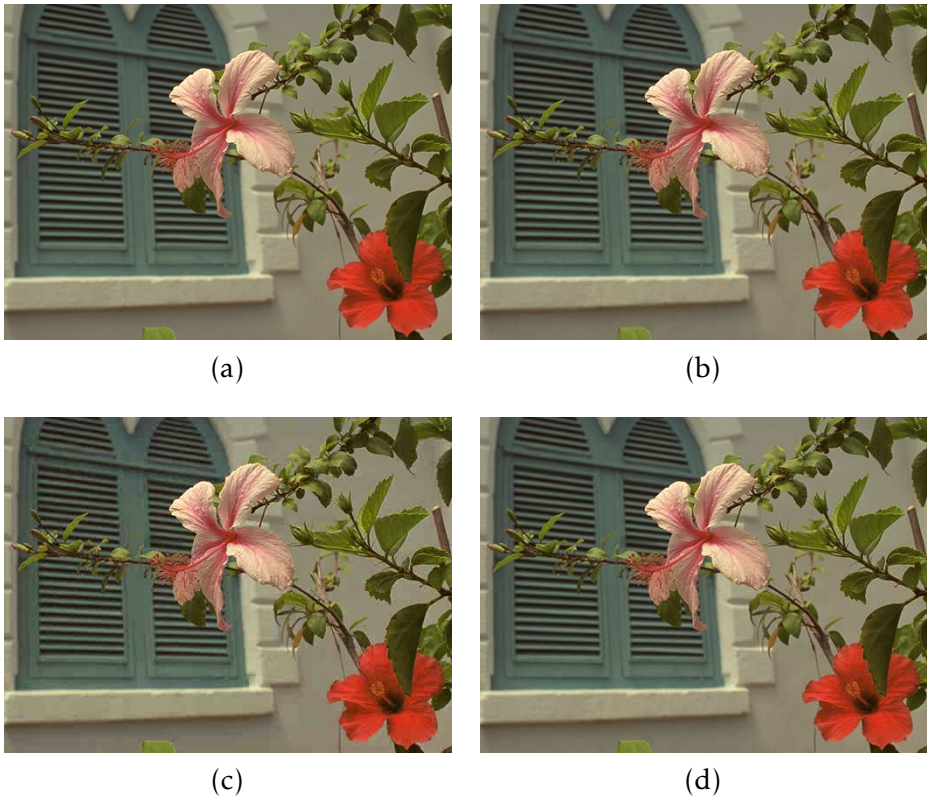
Having found the different quantization tables, we compress the image with both the standard quantization tables, with our modified quantization tables, and with quantization tables consisting of only one power of two, i.e., giving images that visually match the ones obtained by the approach suggested in [JWL11]. For [JWL11], we manually found the power of two resulting in a compressed image of size closest to the standard quantization tables.

Purely ocular evaluation by the author and colleagues could not reliably tell the compressed images apart, see Figure 20. In order to evaluate the similarity of the produced images, we instead use the following four similarity measures to compare compressions of all the images in [PLZ+09] to the uncompressed reference images.

- **MultiScale Structural SIMilarity (MS-SSIM):** Image quality assessment measure intended to match the human perception by moving from a pixel comparison to a structure comparison [WSB03].
- **Feature SIMilarity (FSIM):** Image quality assessment measure intended to match the human perception by using that humans understand pictures mainly by their low-level features. Newer and claimed (by its authors) to be closer to human perception than MS-SSIM [ZZMZ11b]. FSIMc is the color variant of FSIM.
- **Mean Squared Error (MSE):** Classical distance measure not matching the human perception. Found by calculating the mean squared distance between pixels in the images being compared, i.e., when comparing images I_1 and I_2, both of size $M \times N$, we have

$$\mathsf{MSE}(I_1, I_2) = \frac{1}{M \cdot N} \sum_{\substack{1 \le m \le M \\ 1 \le n \le N}} (I_1(m,n) - I_2(m,n))^2. \tag{21}$$

- **Peak Signal Noise Ratio (PSNR):** Classical distance measure not matching the human perception. Derived from the MSE, and taking



(a) (b)

(c) (d)

Figure 20: An image from the TID2008 database [PLZ+09]. In 20a uncompressed, in 20b compressed using the default Q_{50} quantization tables, in 20c using the approach suggested in [JWL11], and in 20d using our approach.

into account the maximal dynamic range of the image. In the same setting as above, and with R being the maximal dynamic range of I_1 and I_2 (so 255 for 8-bit images), we have

$$\mathsf{PSNR}(I_1, I_2) = 10 \cdot \log_{10}\left(\frac{R^2}{\mathsf{MSE}(I_1, I_2)}\right). \tag{22}$$

The procedure described above was implemented in a Matlab script, with calls to the IJG compression implementation [The22]. Functions for calculating MS-SSIM, MSE, and PSNR are provided by Matlab as `multissim`, `imse`, and `psnr`. A Matlab script for calculating FSIM was published as auxiliary material for the article [ZZMZ11b; ZZMZ11a]. Our full code and additional material such as the reference images and compressed variants can be found on author's homepage.[5]

In Table 5, we consider different quality factors ($p = 25$, $p = 50$, and $p = 80$, as described in Section 3.3.2), and compare the average of the images compressed with the standard quantization table with the compressed image obtain from our scheme, and with two images compressed with quantization tables with just one power of two in all entries, as

5 https://serfurth.dk/research/archive/

Table 5: Average results for compressions of the test images from [PLZ+09] compared to the uncompressed images. We started from 3 different quality factors for the unmodified compression (25, 50, and 80), and for each of these we derived the quantization table with only powers of two giving the closest size. For the approach suggested in [JWL11], we include both the closest smaller and the closest larger variant (the value in the quantization tables is indicated in parentheses).

		Size	MS-SSIM	FSIMc	MSE	PSNR
QF25	**Our tables**	16.0 kB	0.960	0.978	77.526	29.749
	Unmodified	15.1 kB	0.959	0.978	78.900	29.655
	[JWL11] (64)	10.4 kB	0.914	0.936	109.016	28.021
	[JWL11] (32)	20.5 kB	0.961	0.976	46.071	31.706
QF50	**Our tables**	25.4 kB	0.979	0.991	45.831	32.008
	Unmodified	24.4 kB	0.979	0.991	45.910	31.988
	[JWL11] (32)	20.5 kB	0.961	0.976	46.071	31.706
	[JWL11] (16)	36.5 kB	0.983	0.992	18.451	35.605
QF80	**Our tables**	43.9 kB	0.990	0.997	20.256	35.402
	Unmodified	43.4 kB	0.991	0.997	20.432	35.364
	[JWL11] (16)	36.5 kB	0.983	0.992	18.451	35.605
	[JWL11] (8)	60.4 kB	0.993	0.997	7.532	39.439

suggest in [JWL11]. We include two images compressed as suggested in [JWL11], since this approach allows so little granularity in the quantization tables that it is often hard to get the compressed size even close to the images compressed with the standard quantization tables. This also demonstrates one of the major issues with the approach suggested in [JWL11]; it only allows 8 different quantization tables, leading to very low granularity. Additionally, as can be seen in Table 5, even when we choose the quantization tables to be the ones resulting in a substantially larger file, the FSIMc score is comparable. Presumably, this is due to all entries in the quantization table having the same value. Thus, this approach does not use the human bias towards noticing low frequency changes.

Comparing the results of the unmodified compression and our compression, Table 5 shows that we can get very close to the the same size and image quality assessment scores (MS-SSIM and FSIMc). In fact, our compression obtain the same FSIMc score as the unmodified, and the MS-SSIM is 0.001 better in one case, and 0.001 worse in one case. This implies that the visual degradation of using our compression is comparable to visual degradation from using the default scheme. The MSE and PSNR values are also similar, but the differences are slightly larger.

We also tested the Lenna test image and images from [AG13] and [Web]. All show essentially the same results.

3.6 FUTURE WORK

This paper extends the theory of digital signatures for images that still allow some form of compression. We present and analyze the first digital signature scheme that is first and foremost designed with the goal of allowing this. Additionally, we have suggested applications where this type of digital signatures could help in a small way towards solving some major societal issues. For these issues, additional work investigating if they would actually help users in a real setting would be a natural next step.

So far, our procedure for generating quantization tables for our compression, has been to fix an image compressed with standard quantization tables, and then finding the tables giving our compressed image size as close to first image as possible. This method is rather inefficient, because it requires many JPEG compressions, among other reasons. Instead of this inefficient method, we suggest defining a new set of standard quantization tables and a function for deriving quantization tables for different quality factors, similar to how it is currently done, but with the new standard and derived tables consisting of only powers of two. These new standard quantization tables should be chosen to preserve the most image quality over a large set of different images and different quality factors, evaluated using both real-life tests with users and image quality assessment measures like the ones used in Section 3.5.

One optimization that could be made to our construction itself, is that chains of hashes that are always truncated by the same number of bits could be combined into one, so that only one hash needs to be provided in the signature for the compressed image. As an example, we refer back to the standard luminance quantization table in Equation (12), in which we see that the table has 14 in entries $(1,2)$, $(2,0)$, and $(3,0)$, and hence the chains of hashes corresponding to these entries will always be truncated by the same amount in our construction. If new quantization tables containing only powers of two are standardized, this optimization would be particular effective, since these tables would necessarily have many identical values. Finally, an efficient prototype should be implemented, and the efficiency of the prototype evaluated and compared to, for example, state of the art of zero-knowledge based approaches [DB23].

4

FOLDING SCHEMES WITH PRIVACY PRESERVING SELECTIVE VERIFICATION

Joan Boyar and Simon Erfurth. Folding schemes with privacy preserving selective verification. *IACR Communications in Cryptology*, 1(4), 2024. URL: https://eprint.iacr.org/2024/1530

ABSTRACT Folding schemes are an exciting new primitive, transforming the task of performing multiple zero-knowledge proofs of knowledge for a relation into performing just one zero-knowledge proof, for the same relation, and a number of cheap inclusion-proofs. Recently, folding schemes have been used to amortize the cost associated with proving different statements to multiple distinct verifiers, which has various applications. We observe that for these uses, leaking information about the statements folded together can be problematic, yet this happens with previous constructions. Towards resolving this issue, we give a natural definition of *privacy preserving* folding schemes, and what security they should offer. To construct privacy preserving folding schemes, we first define *statement hiders*, a primitive which might be of independent interest. In a nutshell, a statement hider hides an instance of a relation as a new instance in the same relation. The new instance is in the relation if and only if the initial instance is. With this building block, we can utilize existing folding schemes to construct a privacy preserving folding scheme, by first hiding each of the statements. Folding schemes allow verifying that a statement was folded into another statement, while statement hiders allow verifying that a statement was hidden as another statement.

4.1 INTRODUCTION

Suppose that N clients outsource some computations to an untrusted server. This server does the computation (possibly with some additional secret data or a proprietary algorithm) and then wishes to prove to the clients that each of their computations was done correctly. One way this could be done is that for each of the N clients, the server provides a (non-interactive) zero-knowledge proof that the client's computation was done correctly. However, this requires doing N (potentially expensive) proofs, one for each of the clients. A *folding scheme* [KST22] allows the server to combine the N statements into just one statement of the same size as the initial statements. Additionally, the folding scheme produces a *folding proof*, which proves that all the statements were folded into the final statement. Thus, the server can prove just the final statement, and distribute the non-interactive zero-knowledge proof of it being correct together with the folding proof to the clients, and all of them should be

convinced that their computations were done correctly. We refer to the server as the *prover* and to each client as a *verifier*.

To be more specific, a folding scheme for an NP-language $\mathcal{L}$ with relation

$$\mathcal{R} = \{(x, w) \mid w \text{ is a proof that } x \in \mathcal{L}\}, \tag{23}$$

can combine instances $(x_i, w_i) \in \mathcal{R}$ for $1 \leq i \leq N$ into one instance $(x, w) \in \mathcal{R}$. Intuitively, (x, w) is in $\mathcal{R}$ if and only if (x_i, w_i) is in $\mathcal{R}$ for $1 \leq i \leq N$. Additionally, a folding scheme produces a folding proof π, that can be used to verify that x was produced by folding the x_i's, even though the verifiers do not necessarily learn w or any of the w_i's.

The folding proof proves that all the statements have been folded into the final statement, and hence, its size is generally $\Omega(N)$. There are two issues with this: (1) Sending a size $\Omega(N)$ folding proof to every verifier is wasteful, if each party only needs to verify that their statement was folded into the final statement. (2) Verifying a folding proof requires knowledge of all statements folded into the final statement, which in a multi-verifier setting has obvious privacy concerns. The first issue has led to the development of *folding schemes with selective verification* [RZ23]. These folding schemes support generating separate proofs of folding for each of the N statements folded together. Each proof of folding π_i should be of size $o(N)$, and need only prove that x_i was folded into the final statement.

However, the folding schemes with selective verification from [RZ23] do not resolve the second issue. Specifically, any folding proof for a specific statement includes either the statement before or the statement after that specific statement. Since each proof of folding is only required to prove that the corresponding statement was folded into the final statement, it is natural to require the proof of folding to preserve the privacy of all other statements, ensuring that no verifier learns anything about the other verifiers' statements.

In this work, we introduce *folding schemes with privacy preserving selective verification*, which resolves both issues (1) and (2). Privacy preserving should be understood as meaning that if a verifier's statement is folded together with other statements, and selective folding proofs are generated and distributed by the prover, other verifiers might learn that the statement is in the language (since the final statement is proven to be in the language), but they will have no idea which statement in the language it is. In practice, we work with an indistinguishability notion, where an adversary chooses two distinct indices i and ℓ and the entire input to the folding scheme, including two potential inputs for the i'th spot. One of the potential i'th inputs is then chosen at random, folding and proof generation is done, and the adversary is then given the statement obtained by folding, and the selective proof of folding for input ℓ. Finally, the adversary has to guess which of the statements was used as the i'th input. We say that the folding scheme is privacy preserving, if no adversary

guesses correctly with probability more than $1/2 + \mathsf{negl}(\lambda)$, where λ is the security parameter.

Toward constructing folding schemes with privacy preserving selective verification, we define a new primitive, which we call an *NP-statement hider*. This primitive is used by the prover, hiding one instance, (x, w), as another instance, (x', w'), and producing a certificate for verifying that x' is hiding x. Using an NP-statement hider and a folding scheme with selective verification as building blocks, we present a generic construction of a folding scheme with privacy preserving selective verification, and show that it satisfies our definition of being privacy preserving. Thus, to extend a folding scheme with selective verification to one with privacy preserving selective verification, it is sufficient to construct a corresponding NP-statement hider. To facilitate this, we present a generic construction of an NP-statement hider, utilizing a folding scheme (which is not required to be privacy preserving). Essentially, the NP-statement hider works by folding the statement to be hidden with a randomly sampled statement. There is evidence that not all folding schemes will allow the required random sampling, but we demonstrate that there is one based on an NP-hard problem that does. Security of the constructed NP-statement hider follows from the security of the underlying folding scheme, and an additional property, which is essentially that an instance hiding one instance is equally likely to hide any other instance. Having this property results in information theoretically hiding NP-statement hiders. We informally state our results in the following theorem.

> THEOREM 4.1 *Combining Theorems 4.10 and 4.12.*
> Let $\mathcal{L}$ be a language with relation $\mathcal{R}$. If there is a folding scheme for $\mathcal{L}$, $\mathcal{R}$ supports efficient sampling of instances, and for any three instances $(x_1, v_1), (x_2, v_2), (x, v) \in \mathcal{R}$ there are as many instances that fold (x_1, v_1) into (x, v) as there are instances folding (x_2, v_2) into (x, v), then there is a folding scheme with privacy preserving selective verification for $\mathcal{L}$.

We apply our constructions to some example folding schemes for algebraic NP-languages. As a warm-up problem, we consider the language *Inner Product Relation of Committed Values* [BCC+16; RZ23]. Then we consider the language *Committed Relaxed R1CS* [KST22], which is also the original language used for folding schemes. We show that both these languages satisfy the conditions of Theorem 4.1.

4.1.1 *Organization of paper*

In Section 4.1.2 we review related work and in Section 4.1.3 we consider possible applications for our work. We review folding schemes in Sections 4.2 and 4.2.1, and folding scheme with selective verification in Section 4.2.2. After this, we define privacy preserving selective verification in Section 4.3 and NP-statement hiders in Section 4.3.1. We construct a privacy preserving folding scheme using an NP-statement hider in a

black-box fashion in Section 4.3.2, and a NP-statement hider using a folding scheme in Section 4.3.3. Finally, in Section 4.4, we apply our constructions to two concrete languages.

4.1.2 *Related Work*

Folding schemes were introduced by Kothapalli, Setty, and Tzialla at CRYPTO'22 [KST22], as a tool to realize incrementally verifiable computation (IVC) [Val08]. IVC, as the name hints, is a method to do computations, such that the correctness of the entire computation can be verified by checking each increment of the computation. Historically, IVC has been constructed using recursive *succinct non-interactive arguments of knowledge* (SNARKs) to prove that each increment was computed correctly. More recently, *accumulators* have been developed [BGH19; BCMS20; BDFG21; BCL+21]. Rather than verifying a SNARK at every increment, an accumulator based scheme allows the SNARK check to be accumulated into the checks from previous increments. At a later time, all steps can be verified by checking a single SNARK, and that the accumulations has been performed correctly at each step. This can be significantly more efficient than checking a SNARK for each step, and communicating the single SNARK and the folding proof requires less communication than communicating a SNARK for each step. The most efficient type of accumulation schemes are folding schemes [NDC+24], and allow one to combine the proofs that each step was computed correctly into one single proof of the same size. Folding schemes yield IVC constructions where the recursive proof that folding (accumulation) was done correctly at each step is dominated by two elliptic curve scalar multiplications, and where the only needed assumption is the discrete logarithm assumption in the random oracle model [KST22]. Nova [KST22] introduced the notion of folding schemes. Since then, folding schemes have attracted much interest for IVC, leading to the development of many folding schemes, for example SuperNova/HyperNova [KS22; KS24a], Protostar [BC23a], LatticeFold [BC24], and Mangrove [NDC+24].

Recently, the *Reductions of Knowledge* framework [KP23; Kot24] was introduced by Kothapalli, one of the authors introducing folding schemes, and Parno. Reductions of knowledge generalizes many flavors of arguments of knowledge, including folding schemes. Generally, a reduction of knowledge reduces checking knowledge of a witness for a statement from one relation, to checking knowledge of a witness for a statement from another (usually simpler) relation. In this framework, a 2-folding scheme for a relation $\mathcal{R}$, is a reduction from $\mathcal{R} \times \mathcal{R}$ to $\mathcal{R}$. Specifically, knowing witnesses (w_1, w_2) to the instance $((x_1, w_1), (x_2, x_2)) \in \mathcal{R} \times \mathcal{R}$ is reduced to knowing witness w to instance $(x, w) \in \mathcal{R}$. While reductions of knowledge do not have folding proofs like folding schemes do, they instead have a requirement that the reduction is *publicly reducible*: given the initial statement(s) and the transcript, any party can reconstruct the final statement. We note that for the folding schemes we consider, the

proof of folding is the first message from the prover, and checking it is exactly reconstructing the final statement.

Ràfols and Zacharakis [RZ23] considers a novel use of folding schemes, by modifying them to allow *selective verification*. Whereas the original version of folding schemes only considers a single verifier verifying all the proofs, and therefore only support verifying that *all* the statements are folded into the final statement, folding schemes with selective verification instead consider multiple verifiers, where each verifier only needs to verify that a subset of the statements are folded into the final statement. Folding schemes with selective verification supports this by generating separate folding proofs for each statement, where each proof only verifies that the matching statement is folded into the final statement. A standard requirement is that each of these proofs should have size sublinear in the total number of statements. Folding schemes with selective verification, are particularly useful in situations where folding proofs are not used as part of incrementally verifiable computations, but rather for verification of delegated computations. Specifically, if many clients outsource their distinct-but-similar computations to a server, and the server has to prove to the clients that it performed the correct computations, it might be more efficient to fold all the proofs into one, rather than separately proving to each client that their computation was done correctly. In this case, rather than sending every client the full folding proof (and all other statements that are folded), the server can send each client only the proof that their statement was folded into the statement that was proven.

Related to folding schemes with selective verification, and hence also to our work, the polynomial commitment scheme, hbPolyCommit, from [YLF+22] uses a Merkle tree structure to amortize the cost of batch processing multiple inner-product arguments, each corresponding to multiple verifiers. The commitment scheme uses the Merkle tree when combining multiple protocol transcripts to produce a challenge. The Merkle tree structure allows each party to verify that their transcript was considered, at a cost that is logarithmic in the number of transcripts. Folding schemes with selective verification differs by considering aggregation of multiple statements into one that is then proved, rather than aggregating multiple proofs together. Similar batch processing of a polynomial commitment scheme is considered in [ZXH+22], but again their work focuses on modifying the proving process, rather than folding statements together.

4.1.3 *Applications*

Folding schemes were initially developed for incrementally verifiable computing, but have since then had multiple other applications. We describe three applications, where folding schemes with privacy preserving selective verification might be useful.

One application, suggested in [RZ23], is for verification in *computation as a service*. Consider a case where many clients (verifiers) delegate similar computations on different inputs to a server (prover). In a trustless setting where interaction is very expensive or impossible, a standard solution is for the prover to use SNARKs to convince the verifiers, that their computations have been performed correctly. The application of folding schemes is straightforward: it amortizes the cost of proving a statement out over all the verifiers' computations, rather than having to prove a statement for each client. Selective verification reduces the communication to each verifier; they need only verify the correctness of their own computations. Privacy preserving selective verification additionally guarantees that the folding proofs do not leak information about other verifiers' computations.

A second application suggested in [RZ23], uses folding proofs with selective verification to share a *verifiable database*. In a verifiable database, clients (verifiers) outsource a database to a server (prover) in a trustless setting. Typically, the verifiers only store a short digest of the database, which allows querying and modifying the database. Viewing the database as a vector, the digest is a homomorphic vector commitment [CF13; CNR+22], querying is simply opening the commitment at a specific location, and modifying is subtracting the original value from the commitment and adding the new one. Rather than opening a commitment for every query, the prover might batch up multiple proofs of opening, fold them together, and then send the SNARK for the folded statement to each verifier with a query in the batch. Again, privacy preserving selective verification both reduces the communication to each verifier and also guarantees that each verifier does not learn what data the other verifiers queried. Privacy preservation would be particularly important in a setting where verifiers might have different privileges, and hence be allowed to access different parts of the database.

Finally, a third application relates to *mitigating the effects of fake news*. While the traditional approach, has been to attempt to flag fake news as such, there has recently been a move towards also flagging authentic content as such. For images, the Adobe lead C2PA initiative [C2P], is currently starting to gain broader adaptation, with both Google and OpenAI having recently joined C2PA. Roughly, the solution proposed by C2PA is to have cameras sign images when they are captured, and then have C2PA compatible programs sign that the edits done to the image are legitimate. When viewing the image, the last signature can be verified, and, ideally, a chain of trust guarantees the authenticity of the image. However, C2PA's approach requires trusting the tools used to edit the image. One approach for resolving this issue, is to use image specific signatures allowing some modifications to be made to the image. However, these signatures come with significant drawbacks, such as supporting only a very limited number of transformations [Erf24], having significant space overheads [JWL11], or only working with rarely used image formats [ZSL04]. Another common approach, is to use zero-knowledge SNARKs to prove

that only certain edits have been applied to an image [NT16]. While this approach is more versatile in which edits it allows, imposes minimal space overhead, and allows efficient verification, it comes with a significant performance costs to the prover. Even the most recent construction takes time on the order of a few minutes to an hour, to generate proofs for a single image [DCB25; MVVZ25]. Here, folding proofs with privacy preserving selective verification could be used to combine the proofs corresponding to many images together, potentially amortizing the cost of proving over many images. For example, suppose that a large (untrusted) social media wishes to support the C2PA approach, but still needs to compress the images uploaded to their platform. Rather than separately proving that each image was compressed by them, they could fold the proofs of many images uploaded in a small time-slot together, using a folding scheme. Selective verification would be sensible, since most likely a user only needs to verify one image at a time. Privacy preservation would be a necessity, since images that are not posted publicly (for example images sent in a private chat) should stay private.

We note that very recently, folding schemes have actually been used to reduce the computation needed to generate a proof that only certain edits have been applied to an image [DEH25]. However, they focus on improving the costs associated with one image, whereas folding schemes with privacy preserving selective verification could be used to amortize this cost out over many images.

4.1.4 *Notation*

Generally, we denote single elements a using lowercase normal weight letters, vectors $\mathbf{a}$ using lowercase bold letters, and matrices A using uppercase normal weight letters. For tuples, we will occasionally be using notation of the form $(x, y, z = (a, b))$. This should be understood as the tuple (x, y, z) where $z = (a, b)$. Similarly, $\{y_i = (a_i, b_i)\}_{1 \le i \le n}$ should be understood as the set $\{y_i\}_{1 \le i \le n}$ where each $y_i = (a_i, b_i)$. For arrows, we use $x \leftarrow_{\$} X$ to denote that x is sampled uniformly from the set X, and $x \leftarrow \mathsf{A}(y)$ to denote that the output of algorithm A on input y is x.

We use additive group notation for cyclic groups, and let $\mathsf{gk} \leftarrow \mathcal{G}(1^\lambda)$ be the description of a group $\mathbb{G}$ over a field $\mathbb{F}$ sampled by a group generation algorithm. A description of a group is $\mathsf{gk} = (\mathbb{G}, \mathcal{P}, p)$, where $\mathbb{G}$ is a finite cyclic group of prime order p and $\mathcal{P}$ is a generator of $\mathbb{G}$. For $\mathcal{P}$ fixed, we denote with $[x]$ the element $x\mathcal{P}$, and let this notation extend naturally to vectors $[\mathbf{v}] \in \mathbb{G}^n$.

4.2 FOLDING SCHEMES

In this section we recall the definition of folding schemes [KST22] and folding schemes with selective verification [RZ23].

As mentioned in the introduction, folding schemes are schemes that allow folding two (or more) NP-statements from a language $\mathcal{L}$ into one

statement from $\mathcal{L}$, crucially of the same size. Intuitively, a statement produced by folding two or more statements, is in $\mathcal{L}$ if and only if all the individual statements are in $\mathcal{L}$. We begin with an informal description of 2-folding schemes, before moving on to a definition of N-folding schemes. Given an NP-language $\mathcal{L}$ and a corresponding relation

$$\mathcal{R} = \{(x, w) \mid w \text{ is a witness for } x \in \mathcal{L}\}, \tag{24}$$

a folding scheme allows efficiently reducing two instances $(x_1, w_1), (x_2, w_2)$ to one instance (x, w). We say that x is obtained by *folding* x_1 and x_2. The folding scheme is also required to output a *folding proof* π that x is the result of folding x_1 and x_2. This proof, together with x, x_1 and x_2, should be a convincing proof that x was formed by folding x_1 and x_2. Similar to standard proofs/arguments of knowledge, folding schemes should essentially have the following properties:

- **Completeness:** If $y_1 = (x_1, w_1) \in \mathcal{R}$ and $y_2 = (x_2, w_2) \in \mathcal{R}$, and folding y_1 and y_2 gives $y = (x, w)$, then $(x, w) \in \mathcal{R}$. Additionally, the folding proof π is accepted.
- **Knowledge soundness:** If (x, w) is the result of folding (x_1, w_1) and (x_2, w_2), and w is a witness that $x \in \mathcal{L}$, then w_i is a witness that $x_i \in \mathcal{L}$ for $i \in \{1, 2\}$.

Following the definition of [RZ23], we formally define N-folding schemes as follows.

Definition 4.2 *N-Folding Scheme.*
For security parameter $\lambda \in \mathbb{N}$, NP-language $\mathcal{L}_{\mathsf{p}}$ parameterized by[a] $\mathsf{p} \leftarrow \mathsf{p}(\lambda)$, $\mathcal{R}_{\mathsf{p}}$ the relation for $\mathcal{L}_{\mathsf{p}}$, and $N = \mathsf{poly}(\lambda)$, an *$N$-folding scheme* FS for the language family $\left\{\mathcal{L}_{\mathsf{p}}\right\}_{\mathsf{p} \leftarrow \mathsf{p}(\lambda)}$ is a tuple of algorithms $(\mathsf{Fold}, \mathsf{FoldVerify})$ which for $n \leq N$ operates as follows.

- $(x, w, \pi) \leftarrow \mathsf{Fold}(\mathsf{p}, (x_1, w_1), \ldots, (x_n, w_n))$. On input parameters p, and n instances $(x_i, w_i) \in \mathcal{R}_{\mathsf{p}}$, Fold outputs an instance (x, w) from $\mathcal{R}_{\mathsf{p}}$ and a folding proof π.
- $0/1 \leftarrow \mathsf{FoldVerify}(\mathsf{p}, x_1, \ldots, x_n, x, \pi)$. On input parameters p, $n+1$ statements $x_1, \ldots, x_n$ and x, and a folding proof π, $\mathsf{FoldVerify}$ outputs 1 if x is the output of folding $x_1, \ldots, x_n$, and 0 otherwise.

Additionally, FS must satisfy the following properties:

- **Completeness:** For all adversaries $\mathcal{A}$

$$\Pr\left[\begin{array}{c} \{y_i\}_{1 \leq i \leq n} \subseteq \mathcal{R}_{\mathsf{p}} \wedge \\ ((x, w) \notin \mathcal{R}_{\mathsf{p}} \vee b = 0) \end{array} \;\middle|\; \begin{array}{c} \{y_i = (x_i, w_i)\}_{1 \leq i \leq n} \leftarrow \mathcal{A}(\mathsf{p}) \\ (x, w, \pi) \leftarrow \mathsf{Fold}(\mathsf{p}, y_1, \ldots, y_n) \\ b \leftarrow \mathsf{FoldVerify}(\mathsf{p}, x_1, \ldots, x_n, x, \pi) \end{array}\right] \leq \mathsf{negl}(\lambda). \tag{25}$$

  Note that we allow $\mathcal{A}$ to be computationally unbounded.

- **Knowledge soundness:** There exists a probabilistic polynomial time (PPT) extractor Ext, such that for all PPT adversaries $\mathcal{A}$

$$\Pr\left[\begin{array}{c|c} (x,w) \notin \mathcal{R}_{\mathsf{p}} \vee b = 0 \vee & (\{x_i\}_{1\le i\le n}, x, w, \pi) \leftarrow \mathcal{A}(\mathsf{p}) \\ \{(x_i,w_i)\}_{1\le i\le n} \subseteq \mathcal{R}_{\mathsf{p}} & b \leftarrow \mathsf{FoldVerify}(\mathsf{p}, x_1, \ldots, x_n, x, \pi) \\ & \{w_i\}_{1\le i\le n} \leftarrow \mathsf{Ext}^{\mathcal{A}}(\mathsf{p}) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda). \quad (26)$$

[a] Here we abuse notation. When writing $\mathsf{p}(\lambda)$ we refer to a (randomized and polynomial time) algorithm that on input the security parameter outputs parameter p.

4.2.1 *Bootstrapping from 2-folding to N-folding*

Generally, folding schemes are constructed as 2-folding schemes, and then turned into N-folding schemes by recursive invocations. Let $\mathsf{FS} = (\mathsf{Fold}, \mathsf{FoldVerify})$ be any 2-folding scheme for a language $\mathcal{L}_{\mathsf{p}}$ with relation $\mathcal{R}_{\mathsf{p}}$. As an example, we construct a 3-folding scheme. Given 3 instances $(x_i, w_i) \in \mathcal{R}$, we fold the three instances into one by first folding two instances into one, and then folding this new instance and the third instance to obtain one final instance:

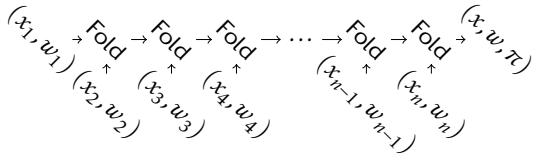
$$(x', w', \pi') \leftarrow \mathsf{Fold}(\mathsf{p}, (x_1, w_1), (x_2, w_2)), \quad (27)$$

$$(x, w, \pi'') \leftarrow \mathsf{Fold}(\mathsf{p}, (x', w'), (x_3, w_3)). \quad (28)$$

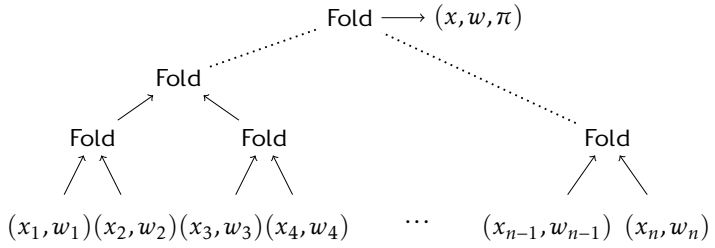
Now the fold of all three instances is $(x, w, \pi = (\pi', \pi''))$. Observe that the folding proof of the 3-folding scheme consists of the folding proofs from both applications of FS.

The essential property making this construction possible, is that the statement generated by a folding scheme is in the same language and of the same size as the original statements. Thus, any 2-folding scheme can immediately be applied in a *bootstrap*-like way to create an N-folding scheme for $N = \mathsf{poly}(\lambda)$. This can be done in many ways, i.e., by chaining the statements together one after the other, or by creating a Merkle tree-like [Mer80; Mer89] structure, see Figures 21a and 21b. Regardless of which approach is used, the folding proof from the N-folding scheme is the accumulated folding proofs from the applications of the 2-folding scheme. Completeness of the N- folding scheme follows immediately from the construction, and observing that an adversary only has a negligible chance of cheating at each step and there are a polynomial number of folds,[1] since $N = \mathsf{poly}(\lambda)$. Knowledge soundness takes more care, but essentially one can construct an extractor by recursively using the extractor for the scheme being bootstrapped. This method results in quasilinear overhead over the extractor for the 2-folding scheme, and once again the probability of extracting valid witnesses is polynomially related to the probability of the extractor for the 2-folding scheme extracting valid witnesses. In [RZ23], the authors give more details on bootstrapping with

1 Both constructions use exactly $N - 1$ folds. The Merkle tree approach has an advantage in that it can naturally be parallelized, both when folding and when verifying



(a) Chaining approach.



(b) Merkle tree approach.

Figure 21: The chaining and Merkle tree approaches to folding n instances (x_i, w_i) into one instance (x, w), using $n-1$ applications of the 2-folding scheme $\mathsf{FS} = (\mathsf{Fold}, \mathsf{FoldVerify})$.

a Merkle tree-like structure, and show that the bootstrapped N-folding scheme is both complete and has knowledge soundness.

4.2.2 *Selective Verification*

The folding scheme verification algorithm from Definition 4.2, takes as input all the folded statements. However, when the number of folded statements is large, this can be very costly if one only wishes to confirm that a single statement x_i was folded into the proven statement x. To solve this issue, [RZ23] introduces folding schemes with selective verification. Rather than having one folding proof π that verifies that all N statements $x_1, \ldots, x_N$ were folded into one statement x, they have N proofs $\pi_1, \ldots, \pi_N$. For each $i \in \{1, \ldots, N\}$, the i'th proof π_i together with x_i and x proves that x_i was folded into x. Note that the size of the proofs should be sublinear in N, since otherwise one could just set $\pi_i = (\pi, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_N)$. Formally, a folding scheme with selective verification is defined as follows.

> Definition 4.3 *Folding Scheme with Selective Verification.*
> For security parameter $\lambda \in \mathbb{N}$, NP-language $\mathcal{L}_{\mathsf{p}}$ parameterized by $\mathsf{p} \leftarrow \mathsf{p}(\lambda)$, $\mathcal{R}_{\mathsf{p}}$ the relation for $\mathcal{L}_{\mathsf{p}}$, $N = \mathsf{poly}(\lambda)$, an N-folding scheme $\mathsf{FS} = (\mathsf{Fold}, \mathsf{FoldVerify})$ for the language family $\left\{\mathcal{L}_{\mathsf{p}}\right\}_{\mathsf{p} \leftarrow \mathsf{p}(\lambda)}$, has *selective verification*, if there is a tuple of algorithms $(\mathsf{SlctProve}, \mathsf{SlctVerify})$ which for $n \leq N$ operates as follows.

- $(\pi_1,\dots,\pi_n) \leftarrow \mathsf{SlctProve}(\mathsf{p}, x_1,\dots,x_n, x, \pi)$. On input parameters p, $n+1$ statements $x_1,\dots,x_n$ and x, and folding proof π, SlctProve outputs proofs $\pi_1,\dots,\pi_n$.
- $0/1 \leftarrow \mathsf{SlctVerify}(\mathsf{p}, x, i, x_i, \pi_i)$. On input parameters p, statements x and x_i, integer $i \in \{1,\dots,n\}$, and folding proof π_i, SlctVerify outputs 1 if x_i is folded into x.

Additionally, the following properties must be satisfied.

- **Selective completeness:** For all adversaries $\mathcal{A}$

$$\Pr\left[\begin{array}{c} \{y_i\}_{1\le i\le n} \subseteq \mathcal{R}_{\mathsf{p}} \\ \wedge\ (1 \le j \le n) \\ \wedge\ b = 0 \end{array} \;\middle|\; \begin{array}{c} (\{y_i = (x_i, w_i)\}_{1\le i\le n}, j) \leftarrow \mathcal{A}(\mathsf{p}) \\ (x, w, \pi) \leftarrow \mathsf{Fold}(\mathsf{p}, y_1,\dots,y_n) \\ (\pi_1,\dots,\pi_n) \leftarrow \mathsf{SlctProve}(\mathsf{p}, x_1,\dots,x_n, x, \pi) \\ b \leftarrow \mathsf{SlctVerify}(\mathsf{p}, x, j, x_j, \pi_j) \end{array}\right] \le \mathsf{negl}(\lambda). \tag{29}$$

- **Selective knowledge soundness:** There exists a PPT extractor Ext such that for all PPT adversaries $\mathcal{A}$

$$\Pr\left[\begin{array}{c} (x, w) \notin \mathcal{R}_{\mathsf{p}} \vee b = 0 \vee \\ (x_i, w_i) \in \mathcal{R}_{\mathsf{p}} \end{array} \;\middle|\; \begin{array}{c} (i, x_i, \pi_i, x, w) \leftarrow \mathcal{A}(\mathsf{p}) \\ b \leftarrow \mathsf{SlctVerify}(\mathsf{p}, x, i, x_i, \pi_i) \\ w_i \leftarrow \mathsf{Ext}^{\mathcal{A}}(\mathsf{p}) \end{array}\right] \ge 1 - \mathsf{negl}(\lambda). \tag{30}$$

- **Efficiency:** The size of each π_i is sublinear in n, i.e., $|\pi_i| = o(n)$.

Remark 4.4
A folding scheme can be equipped with selective verification as follows. First, the Merkle tree-like bootstrapping construction, illustrated in Figure 21b, is used to get an N-folding scheme from a 2-folding scheme. Each π_i consists of the folding proofs for the 2-folding schemes used on the path between x_i and x, together with the statements these 2-folding schemes take as input, that are not already on the path between x_i and x.

For example, either x_{i-1} or x_{i+1} will be in π_i, since one of these is input to the 2-folding scheme taking x_i as input. However, x_i will not be in π_i, since it is on the path between x_i and x.

When this approach is used to make a folding scheme satisfying Definition 4.2 selectively verifiable, it follows relatively straightforwardly that the bootstrapped construction satisfies Definition 4.3. Selective completeness follows immediately from the construction, and selective knowledge soundness from an argument similar to the argument for the bootstrapped construction having knowledge soundness, except that one now only need to follow one path from the root to x_i. Once again, more details can be found in [RZ23], where full algorithms for SlctProve and

SlctVerify can also be found. For efficiency, it is easily observed that the path between x_i and x has length $O(\log n)$, and that each instance of the 2-folding scheme along the path results in 1 folding proof and requires 1 extra statement. Since both of these are constant sized with respect to the number of statements folded together, the size of π_i is $O(\log N)$, and hence sublinear in N. The notion of selective verification can be generalized to folding proofs for subsets of $\{x_i, \ldots, x_N\}$. From the Merkle tree literature [BELN23], it follows that in the case where one allows arbitrary subsets of the x_i, the number of additional statements one needs to provide might be linear in N, but if one requires the subset to be consecutive statements, the number of additional statements is still guaranteed to be logarithmic in N. However, one would still need to provide up to $2N - 1$ folding proofs from the underlying 2-folding scheme.

4.3 PRIVACY PRESERVING SELECTIVE VERIFICATION

The original definition of folding schemes has no notion of a folding scheme being "private", which makes sense since knowledge of all x_i's folded into x is required to verify the folding proof for x. Thus, there is in some sense nothing to be kept private, but the witnesses. However, for folding schemes with selective verification it seems natural to define *privacy preserving selective verification*, which, informally, extends Definition 4.3 with a guarantee that a folding proof π_i for x_i does not leak information about x_j for $j \neq i$. It can immediately be observed that the folding scheme with selective verification described in Remark 4.4 is not privacy preserving; any π_i includes either x_{i-1} or x_{i+1}. In this section, we first define *folding schemes with privacy preserving selective verification*, and then discuss a general approach to making folding schemes with selective verification privacy preserving, using a generic mechanism for hiding NP statements, which we formally define in Definition 4.6. This results in Construction 4.7 and Theorem 4.10. It is possible to construct hiding mechanisms for (some) NP-languages from folding schemes, which we do in Construction 4.11 and Theorem 4.12. In Section 4.4, we give examples of folding schemes with privacy preserving selective verification, using the mechanisms from this section.

At the core of our definition of a folding scheme being privacy preserving, is a notion of indistinguishability under chosen-message attack. In our definition, we allow an adversary to choose an index j it wishes to attack, an index $\ell \neq j$ for which it will get the proof π_ℓ, the (valid) inputs (x_i, w_i) for all indices $i \neq j$, and two (valid) potential inputs (x_j^0, w_j^0) and (x_j^1, w_j^1) for j. For random $b \leftarrow_\$ \{0,1\}$, folding is then done with (x_j^b, w_j^b), and the selective verification folding proofs are generated. Finally, the adversary is given x and π_ℓ, and has to guess b. For simplicity, we use a pair of algorithms as the adversary, but allow passing information from the first algorithm to the second through a state s. A folding scheme has privacy preserving selective verification if the probability of any adversary

guessing correctly is only negligibly better than $\frac{1}{2}$. We formally define this in Definition 4.5.

DEFINITION 4.5 *Folding Schemes with Privacy Preserving Selective Verification.*
For security parameter $\lambda \in \mathbb{N}$, NP-language $\mathcal{L}_{\mathsf{p}}$ parameterized by $\mathsf{p} \leftarrow \mathsf{p}(\lambda)$, $\mathcal{R}_{\mathsf{p}}$ the relation for $\mathcal{L}_{\mathsf{p}}$, $N = \mathsf{poly}(\lambda)$, an N-folding scheme with selective verification $\mathsf{FS} = (\mathsf{Fold}, \mathsf{FoldVerify}, \mathsf{SlctProve}, \mathsf{SlctVerify})$ for the language family $\left\{\mathcal{L}_{\mathsf{p}}\right\}_{\mathsf{p}\leftarrow\mathsf{p}(\lambda)}$, is said to be a *folding scheme with privacy preserving selective verification* if for $n \leq N$ and all adversaries $\mathcal{A}$ consisting of a pair of algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$,

$$\Pr\left[\begin{array}{c} \{y_i\}_{\substack{1\leq i\leq n\\ i\neq j}} \subseteq \mathcal{R}_{\mathsf{p}} \wedge \\ \{(x_j^0, w_j^0), (x_j^1, w_j^1)\} \subseteq \mathcal{R}_{\mathsf{p}} \\ \wedge\ \ell \neq j\ \wedge\ b' = b \end{array} \middle| \begin{array}{c} \left(\begin{array}{c} \ell, j, \{y_i = (x_i, w_i)\}_{\substack{1\leq i\leq n\\ i\neq j}}, \\ (x_j^0, w_j^0), (x_j^1, w_j^1), s \end{array}\right) \leftarrow \mathcal{A}_1(\mathsf{p}) \\ b \leftarrow_{\$} \{0,1\} \\ (x, w, \pi) \leftarrow \mathsf{Fold}(\mathsf{p}, y_1, \ldots, (x_j^b, w_j^b), \ldots, y_n) \\ (\pi_1, \ldots, \pi_n) \leftarrow \mathsf{SlctProve}\left(\begin{array}{c} \mathsf{p}, x_1, \ldots, x_j^b, \\ \ldots, x_n, x, \pi \end{array}\right) \\ b' \leftarrow \mathcal{A}_2(\mathsf{p}, x, \ell, x_\ell, \pi_\ell, s) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda). \tag{31}$$

4.3.1 *NP-statement hider*

As previously mentioned, the construction from Remark 4.4 does not satisfy Definition 4.5. However, we observe that if the prover somehow "hides" the statements before folding them, we can reuse the construction. This motivates the following definition of a hiding mechanism, which on an instance $(x, w) \in \mathcal{R}$ and randomness $r \in \mathfrak{R}$ "hides" (x, w) as $(x', w') \in \mathcal{R}$. The hiding mechanism also outputs a certificate c, which can be used to verify that x' is hiding x. This certificate could, for example, include the randomness r. With such a mechanism, it is straightforward to get a folding scheme with privacy preserving selective verification. First, each instance is hidden, then all the hidden instances are folded, and finally the selective folding proofs π_i are updated to also include c_i. Crucially, π_i includes neither x_{i-1} nor x_{i+1}, but rather x'_{i-1} or x'_{i+1}, which, assuming the hiding mechanism is secure, do not reveal the original statements. We formalize this construction in Construction 4.7, but first formally define hiding mechanisms.

DEFINITION 4.6 *NP-Statement Hider.*
For security parameter $\lambda \in \mathbb{N}$ and NP-language $\mathcal{L}_{\mathsf{p}}$ parameterized by $\mathsf{p} \leftarrow \mathsf{p}(\lambda)$, with relation $\mathcal{R}_{\mathsf{p}}$, an NP-statement hider for $\mathcal{L}_{\mathsf{p}}$ is a pair of

efficient algorithms $\mathsf{SH} = (\mathsf{Hide}, \mathsf{Check})$ such that for $(x, w) \in \mathcal{R}_\mathsf{p}$ and random string $r \in \mathfrak{K}$ from randomness space $\mathfrak{K}$, SH acts as follows:

- $(x', w', c) \leftarrow \mathsf{Hide}(\mathsf{p}, x, w, r)$ on input parameters p, instance $(x, w) \in \mathcal{R}_\mathsf{p}$ and randomness $r \in \mathfrak{K}$, Hide outputs $(x', w') \in \mathcal{R}_\mathsf{p}$ and certificate c.
- $0/1 \leftarrow \mathsf{Check}(\mathsf{p}, x, x', c)$ on input parameters p, statements $\{x, x'\} \subseteq \mathcal{L}_\mathsf{p}$, and certificate c, Check outputs 1 if the certificate shows that x' is hiding x.

Additionally, SH must satisfy the following properties.

- Completeness: For all adversaries $\mathcal{A}$

$$\Pr\left[\begin{array}{c|c} (x,w) \in \mathcal{R}_\mathsf{p} \wedge & (x,w) \leftarrow \mathcal{A}(\mathsf{p}) \\ ((x',w') \notin \mathcal{R}_\mathsf{p} \vee b = 0) & r \leftarrow_\$ \mathfrak{K} \\ & (x',w',c) \leftarrow \mathsf{Hide}(\mathsf{p},x,w,r) \\ & b \leftarrow \mathsf{Check}(\mathsf{p},x,x',c) \end{array}\right] \leq \mathsf{negl}(\lambda). \tag{32}$$

- Knowledge soundness: There exists a PPT extractor Ext, such that for all PPT adversaries $\mathcal{A}$

$$\Pr\left[\begin{array}{c|c} x \notin \mathcal{L}_\mathsf{p} \vee (x',w') \notin \mathcal{R}_\mathsf{p} & (x,x',w',c) \leftarrow \mathcal{A}(\mathsf{p}) \\ \vee\, b = 0 \vee (x,w) \in \mathcal{R}_\mathsf{p} & b \leftarrow \mathsf{Check}(\mathsf{p},x,x',c) \\ & w \leftarrow \mathsf{Ext}^{\mathcal{A}}(\mathsf{p}) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda). \tag{33}$$

- Hiding: For all adversaries $\mathcal{A}$ consisting of a pair of algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$,

$$\Pr\left[\begin{array}{c|c} \{(x_0,w_0),(x_1,w_1)\} \subseteq \mathcal{R}_\mathsf{p} & (x_0,w_0,x_1,w_1,s) \leftarrow \mathcal{A}_1(\mathsf{p}) \\ \wedge\, b' = b & b \leftarrow_\$ \{0,1\}, r \leftarrow_\$ \mathfrak{K} \\ & (x',w',c) \leftarrow \mathsf{Hide}(\mathsf{p},x_b,w_b,r) \\ & b' \leftarrow \mathcal{A}_2(x',w',s) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda). \tag{34}$$

4.3.2 *Privacy preserving folding scheme from an NP-statement hider*

In Construction 4.7, we construct a folding scheme with privacy preserving selective verification, using an NP-statement hider and a folding scheme with selective verification as building blocks. We show that the construction is secure in Theorem 4.10. At a high level, this construction does exactly what we previously described: each statement is hidden, then the statements hiding the original statements are folded together, and, finally, all proofs are updated to include some additional

information, allowing checking that the statements hiding the original statements do indeed hide the statements they are claimed to be hiding. Similarly, the verification algorithms both check that the hiding(s) are as claimed, and that the folding is correct.

CONSTRUCTION 4.7 PrivateFS.
Let $\mathsf{SH} = (\mathsf{SH.Hide}, \mathsf{SH.Check})$ be an NP-statement hider and $\mathsf{FS} = (\mathsf{FS.Fold}, \mathsf{FS.FoldVerify}, \mathsf{FS.SlctProve}, \mathsf{FS.SlctVerify})$ be a folding scheme with selective verification. Then

$$\mathsf{PrivateFS} = (\mathsf{Fold}, \mathsf{FoldVerify}, \mathsf{SlctProve}, \mathsf{SlctVerify}),$$

constructed as follows, is a folding scheme with privacy preserving selective verification.

- $\mathsf{Fold}(\mathsf{p}, (x_1, w_1), \ldots, (x_n, w_n))$:
  1. Generate randomness $r_1, \ldots, r_n \in \mathfrak{R}$.
  2. For $1 \le i \le n$: $(x_i', w_i', c_i) \leftarrow \mathsf{SH.Hide}(\mathsf{p}, x_i, w_i, r_i)$.
  3. $(x, w, \pi') \leftarrow \mathsf{FS.Fold}(\mathsf{p}, (x_1', w_i'), \ldots (x_n', w_n'))$.
  4. Output $(x, w, \pi = (\pi', (c_1, x_1'), \ldots, (c_n, x_n')))$.
- $\mathsf{FoldVerify}(\mathsf{p}, x_1, \ldots, x_n, x, \pi)$:
  1. Parse π as $(\pi', (c_1, x_1'), \ldots, (c_n, x_n'))$.
  2. For $1 \le i \le n$: if $\mathsf{SH.Check}(\mathsf{p}, x_i, x_i', c_i) = 0$, output 0 and abort.
  3. If $\mathsf{FS.FoldVerify}(\mathsf{p}, x_1', \ldots, x_n', x, \pi') = 0$, output 0 and abort.
  4. Output 1.
- $\mathsf{SlctProve}(\mathsf{p}, x_1, \ldots, x_n, x, \pi)$:
  1. Parse π as $(\pi', (c_1, x_1'), \ldots, (c_n, x_n'))$.
  2. $(\pi_1', \ldots, \pi_n') \leftarrow \mathsf{FS.SlctProve}(\mathsf{p}, x_1', \ldots, x_n', x, \pi')$.
  3. Output $\left(\pi_i = (\pi_i', c_i, x_i')\right)_{1 \le i \le n}$.
- $\mathsf{SlctVerify}(\mathsf{p}, x, i, x_i, \pi_i)$:
  1. Parse π_i as (π_i', c_i, x_i').
  2. If $\mathsf{SH.Check}(\mathsf{p}, x_i, x_i', c_i) = 0$, output 0 and abort.
  3. If $\mathsf{FS.SlctVerify}(\mathsf{p}, x, i, x_i', \pi_i') = 0$, output 0 and abort.
  4. Output 1.

REMARK 4.8
It is immediate that the modifications in Construction 4.7 do not affect the asymptotic efficiency of the underlying folding scheme. In particular, the size of each proof π_i only grows by a constant amount in n.

REMARK 4.9
Note that if a folding scheme with privacy preserving selective verification is used in a situation where it is frequent that a verifier will have

to verify more than one proof of folding, generating the randomness r_1 to r_n with a seed-tree [BKP20] can result in less communication to each verifier. If a verifier has to verify the folding proofs of multiple consecutive statements, the randomness included in each of the folding proofs can often be replaced with fewer seeds from levels closer to the root of the seed-tree.

THEOREM 4.10
If SH is an NP-statement hider satisfying Definition 4.6 and FS is a folding scheme with selective verification satisfying Definition 4.3, then PrivateFS from Construction 4.7 is a folding scheme with privacy preserving selective verification, in the sense of Definition 4.5.

Proof. We must show that

$$\mathsf{PrivateFS} = (\mathsf{Fold}, \mathsf{FoldVerify}, \mathsf{SlctProve}, \mathsf{SlctVerify}), \tag{35}$$

as constructed in Construction 4.7, has the properties described in Definitions 4.2, 4.3 and 4.5. For brevity, we show selective completeness, selective knowledge soundness, and privacy preserving, i.e., the properties explicitly outlined in Definitions 4.3 and 4.5. Completeness and knowledge soundness (Definition 4.2) follows from very similar arguments.[2]

SELECTIVE COMPLETENESS follows from showing that an adversary against PrivateFS's selective completeness implies an adversary against either the selective completeness of FS or the completeness of SH. Recall from Definition 4.3 that an adversary $\mathcal{A}$ against PrivateFS's selective completeness chooses a valid input $\{(x_i, w_i)\}_{1 \le i \le n}$ to PrivateFS.Fold and an index j, trying to make $b = 0$, where b is given by

$$(x, w, \pi) \leftarrow \mathsf{PrivateFS.Fold}(\mathsf{p}, y_1, \ldots, y_n) \tag{36}$$

$$(\pi_1, \ldots, \pi_n) \leftarrow \mathsf{PrivateFS.SlctProve}(\mathsf{p}, x_1, \ldots, x_n, x, \pi) \tag{37}$$

$$b \leftarrow \mathsf{PrivateFS.SlctVerify}(\mathsf{p}, x, j, x_j, \pi_j). \tag{38}$$

From Construction 4.7, it is clear that if $b = 0$, then either $\mathsf{SH.Check}(\mathsf{p}, x_i, x_i', c_i) = 0$ or $\mathsf{FS.SlctVerify}(\mathsf{p}, x, i, x_i', \pi_i') = 0$.

Consider first if $\mathsf{SH.Check}(\mathsf{p}, x_i, x_i', c_i) = 0$. Since $(x_i, w_i) \in \mathcal{R}_\mathsf{p}$, the randomness $r_i \in \mathfrak{R}$ was chosen at random, and x_i' and c_i generated as $(x_i', w_i', c_i) \leftarrow \mathsf{SH.Hide}(\mathsf{p}, x_i, w_i, r_i)$, we are in exactly the situation described by SH's completeness definition (Definition 4.6). Thus, in this case, $\mathcal{A}$ being successful implies an adversary against SH being complete.

On the other hand, if $\mathsf{FS.SlctVerify}(\mathsf{p}, x, i, x_i', \pi_i') = 0$, we observe that each (x_i', w_i') is in $\mathcal{R}_\mathsf{p}$ (otherwise, we again have an adversary to SH being complete), and thus we are now in exactly the situation described by

2 The main difference is that for selective completeness and selective knowledge soundness, we look at just one specific index of the input (and the output of hiding it). For completeness and knowledge soundness, however, we have to look at all inputs (either when finding an adversary to FS or SH, or when constructing an extractor against PrivateFS). Since the number of inputs is polynomial in n, the constructed adversaries and extractor are still polynomial time.

FS's selective completeness definition (Definition 4.3), where n instances (x'_i, w'_i) are first folded together using FS.Fold, and selective proofs are then generated using FS.SlctProve. Thus, if $\mathsf{FS.SlctVerify}(\mathsf{p}, x, i, x'_i, \pi'_i) = 0$, we see that again $\mathcal{A}$ implies an adversary to either SH being complete or to FS being selective complete.

SELECTIVE KNOWLEDGE SOUNDNESS can be shown by constructing an extractor Ext for adversaries against PrivateFS's selective knowledge soundness, using the selective knowledge soundness extractor FS.Ext for FS, and the knowledge soundness extractor SH.Ext for SH. Essentially, we first use FS.Ext to extract a witness for x'_i, and then SH.Ext to extract a witness for x_i.

Assume that an adversary $\mathcal{A}$ against the selective knowledge soundness of PrivateFS outputs (i, x_i, π_i, x, w), where π_i can be parsed as $\pi_i = (\pi'_i, c_i, x'_i)$. We construct Ext as follows.

1. To extract w'_i such that $(x'_i, w'_i) \in \mathcal{R}_\mathsf{p}$, create an adversary $\mathcal{A}_\mathsf{FS}$, which itself queries $\mathcal{A}$, but then outputs (i, x'_i, π'_i, x, w). Note that if $\mathcal{A}$ is successful against PrivateFS, then $\mathcal{A}_\mathsf{FS}$ is successful against FS, since PrivateFS.SlctVerify invokes FS.SlctVerify.

2. Ext invokes FS.Ext with access to $\mathcal{A}_\mathsf{FS}$, obtaining w'_i.

3. To extract w_i such that $(x_i, w_i) \in \mathcal{R}_\mathsf{p}$, create an adversary $\mathcal{A}_\mathsf{SH}$ which queries $\mathcal{A}$, extracts w'_i using FS.Ext, and then outputs (x_i, x'_i, w'_i, c_i). Similarly to $\mathcal{A}_\mathsf{FS}$, we see that if $\mathcal{A}$ is successful against PrivateFS, then $\mathcal{A}_\mathsf{SH}$ is successful against SH.

4. Ext invokes SH.Ext with access to $\mathcal{A}_\mathsf{SH}$, obtaining w_i, which Ext then outputs.

Since $\mathcal{A}_\mathsf{FS}$ and $\mathcal{A}_\mathsf{SH}$ are successful if $\mathcal{A}$ is successful, both FS.Ext and SH.Ext are successful with overwhelming probability if $\mathcal{A}$ is successful, and, hence, so is Ext.

PRIVACY PRESERVATION follows from showing that an adversary against PrivateFS's privacy preserving property implies an adversary against SH being hiding, similarly to how selective completeness was shown. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary against PrivateFS's privacy preserving property (Definition 4.5). We now construct an adversary $\mathsf{SH}.\mathcal{A} = (\mathsf{SH}.\mathcal{A}_1, \mathsf{SH}.\mathcal{A}_2)$ against SH being hiding as follows.

- $\mathsf{SH}.\mathcal{A}_1$: Run $\mathcal{A}_1$ to get $(\ell, j, \{(x_i, w_i)\}_{\substack{1 \le i \le n \\ i \neq j}}, (x_j^0, w_j^0), (x_j^1, w_j^1), s_{\mathsf{PrivateFS}})$. Output $(x_j^0, w_j^0, x_j^1, w_j^1, s)$, where $s = (\ell, j, \{(x_i, w_i)\}_{\substack{1 \le i \le n \\ i \neq j}}, s_{\mathsf{PrivateFS}})$ is the state passed on to $\mathsf{SH}.\mathcal{A}_2$.

- $\mathsf{SH}.\mathcal{A}_2$: On input (x', w', s), hiding either (x_j^0, w_j^0) or (x_j^1, w_j^1), essentially emulate PrivateFS.Fold and PrivateFS.SlctProve, to generate correct input to $\mathcal{A}_2$.

1. For $i \neq j$, hide (x_i, w_i) using SH.Hide with randomness $r_i \leftarrow_\$ \mathfrak{K}$, obtaining (x_i', w_i', c_i). Then, fold all hidden statements using FS.Fold with (x', w') in the j'th spot;

$$(x, w, \pi') \leftarrow \mathsf{FS.Fold}(\mathsf{p}, (x_1', w_1'), \ldots, (x', w'), \ldots, (x_n', w_n')). \quad (39)$$

2. Next, run $\mathsf{FS.SlctProve}(\mathsf{p}, x_1', \ldots, x_{j-1}', x', x_{j+1}', \ldots, x_n', x, \pi')$ to get $(\pi_1', \ldots, \pi_n')$. Generate π_ℓ by joining π_ℓ' and (c_ℓ, x_ℓ'). Since $\ell \neq j$, these are known.
3. Run $\mathcal{A}_2(\mathsf{p}, x, \ell, x_\ell, \pi_\ell, s_{\mathsf{PrivateFS}})$ to obtain b'. Output b'.

Observe that the input to $\mathcal{A}_2$ is *exactly* the same if $\mathcal{A}_2$ is running directly on PrivateFS, since c_j is not part of the input, and is not used for generating any of the input, besides $(x_j', w_j') = (x', w')$, which is still generated using randomness sampled from $\mathfrak{K}$. Thus, $(\mathsf{SH}.\mathcal{A}_1, \mathsf{SH}.\mathcal{A}_2)$ is successful exactly when $(\mathcal{A}_1, \mathcal{A}_2)$ is successful, and hence it follows from the assumption that SH is hiding that PrivateFS is privacy preserving. □

Note that this proof also show that the scheme is private against colluding adversaries, i.e., with respect to a stronger version of Definition 4.5, where $\mathcal{A}_2$ is not only given π_ℓ, but instead all folding proofs except π_j. The only change to the proof would be that $\mathsf{SH}.\mathcal{A}_2$ queries $\mathcal{A}_2$ with $n-1$ folding proofs, instead of just π_ℓ.

4.3.3 *NP-statement hider from a folding scheme*

At this point, we have a folding scheme with selective verification from [RZ23], and we know that a folding scheme with selective verification together with an NP-statement hider is enough to give us a folding scheme with privacy preserving selective verification. Thus, the next question we ask is how to construct an NP-statement hider? One straightforward approach, is to hide an instance (x, w) by folding it with another instance $(x_\$, w_\$)$, using the folding scheme for the language, producing a new instance (x', w'), which is used as the output of the statement hider. The certificate c will then be the folding proof of folding (x, w) and $(x_\$, w_\$)$, together with either $x_\$$, or the seed used to generate $(x_\$, w_\$)$. Then, checking that x' is hiding x is just verifying that x was folded into x'.

To show that an NP-statement hider for a language, $\mathcal{L}$, with relation, $\mathcal{R}$, constructed in this fashion is secure, it is roughly sufficient that two properties hold: Let $\mathcal{R}' \subset \mathcal{R}$. We require that for any two instances $(x_0, w_0), (x_1, w_1) \in \mathcal{R}$ and any $(x_\$, w_\$) \in \mathcal{R}'$, there exists $(x_\$', w_\$') \in \mathcal{R}'$ such that

$$\mathsf{Fold}((x_0, w_0), (x_\$, w_\$)) = (x, w) = \mathsf{Fold}((x_1, w_1), (x_\$', w_\$')), \quad (40)$$

where we abuse notion and ignore the folding proof. For any (x, w) there should also be as many ways to hide (x_0, w_0) as (x, w), as there are ways to hide (x_1, w_1) as (x, w). In addition, we require that efficient sampling random instances from $\mathcal{R}'$ is possible, and we select $(x_\$, w_\$)$ randomly from

$\mathcal{R}'$. If these properties hold, it is straightforward to see that since $(x_\$, w_\$)$ is sampled from $\mathcal{R}'$, it is equally likely that $(x'_\$, w'_\$)$ is sampled. Thus, (x, w) is just as likely to hide (x_1, w_1) as (x_0, w_0), and hence no adversary can do better than random guessing, showing hiding. Completeness and soundness follow directly from the underlying folding scheme. We present the construction of an NP-statement hider from a folding scheme in Construction 4.11, and show that it is secure in Theorem 4.12.

CONSTRUCTION 4.11 *NP-statement hider from folding.*
Let $\mathsf{FS} = (\mathsf{Fold}, \mathsf{FoldVerify})$ be a folding scheme for a language $\mathcal{L}$ with relation $\mathcal{R}$, and $\mathcal{R}' \subseteq \mathcal{R}$ a subset used as the random space $\mathfrak{R}$. That is, Hide takes a random instance $(x_\$, w_\$) \in \mathcal{R}'$ as its randomness input. Then, $\mathsf{SH} = (\mathsf{Hide}, \mathsf{Check})$, constructed as follows, is an NP-statement hider.

- $\mathsf{Hide}(\mathsf{p}, x, w, (x_\$, w_\$))$
  1. Fold (x, w) and $(x_\$, w_\$)$ together:
  $$(x', w', \pi) \leftarrow \mathsf{FS.Fold}(\mathsf{p}, (x, w), (x_\$, w_\$)) \tag{41}$$
  2. Output (x', w', c) where $c = (x_\$, \pi)$.
- $\mathsf{Check}(\mathsf{p}, x, x', c)$
  1. Parse c as $(x_\$, \pi)$.
  2. Output the result of $\mathsf{FS.FoldVerify}(\mathsf{p}, x, x_\$, x', \pi)$.

THEOREM 4.12
Suppose FS satisfies Definition 4.2, $\mathcal{R}'$ allows efficient sampling, and

1. For any two instances $(x_0, w_0), (x_1, w_1) \in \mathcal{R}$ and any $(x_\$, w_\$) \in \mathcal{R}'$, there exists $(x'_\$, w'_\$) \in \mathcal{R}'$ such that the outputs of $\mathsf{Fold}((x_0, w_0), (x_\$, w_\$))$ and $\mathsf{Fold}((x_1, w_1), (x'_\$, w'_\$))$ agree everywhere, except on the folding proofs.
2. For any $(x_0, w_0), (x_1, w_1), (x, w) \in \mathcal{R}$, if $\{H_i\}_{i \in \{0,1\}}$ are the sets of elements $(x_\$, w_\$) \in \mathcal{R}'$ such that hiding (x_i, w_i) with $(x_\$, w_\$)$ results in (x, w), then $|H_0| = |H_1|$.

Then, the cryptographic scheme defined in Construction 4.11 is an NP-statement hider satisfying Definition 4.6.

Proof. Completeness and knowledge soundness follow from FS satisfying Definition 4.2 for 2 statements. We give outlines for how they are argued.

For completeness, an adversary $\mathsf{FS}.\mathcal{A}$ against FS being complete, in the sense of Definition 4.2, can be constructed from an adversary $\mathcal{A}$ against Construction 4.11 being complete in the sense of Definition 4.6. $\mathsf{FS}.\mathcal{A}$ invokes $\mathcal{A}$ to get (x, w) and samples a random instance $(x_\$, w_\$) \in \mathfrak{R}$. It then outputs $((x, w), (x_\$, w_\$))$. By inspecting Hide and Check, it can be confirmed that from this point on, everything is computed the same way in FS's completeness definition and in Construction 4.11's completeness definition, and $\mathsf{FS}.\mathcal{A}$ is successful if $\mathcal{A}$ is successful.

For knowledge soundness, the extractor FS.Ext implied by FS having knowledge soundness can be used to construct an extractor Ext for Construction 4.11's knowledge soundness. To do this, we create an adversary FS.$\mathcal{A}$ that FS.Ext queries. FS.$\mathcal{A}$ runs $\mathcal{A}$ to get (x, x', w', c) and uses the information in c to derive $x_\$$, which was used to hide x. Finally, FS.$\mathcal{A}$ outputs $(x, x_\$, x', w', \pi)$. Now, Ext runs $\mathsf{FS.Ext}^{\mathsf{FS}.\mathcal{A}}$, to obtain and output w.

For hiding, let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be any adversary. Consider the (x_0, w_0) and (x_1, w_1) output from $\mathcal{A}_1$. For $b \in \{0,1\}$ and $(x_\$, w_\$) \in \mathfrak{R}$, both sampled at random, $\mathcal{A}_2$ receives (x', w') where

$$(x', w', c) \leftarrow \mathsf{Hide}(\mathsf{p}, x_b, w_b, (x_\$, w_\$)) = \mathsf{FS.Fold}(\mathsf{p}, (x, w), (x_\$, w_\$)). \quad (42)$$

The requirements of the theorem imply that there is $(x'_\$, w'_\$) \in \mathcal{R}' = \mathfrak{R}$ such that

$$\mathsf{Fold}((x_b, w_b), (x_\$, w_\$)) = \mathsf{Fold}((x_{1-b}, w_{1-b}), (x'_\$, w'_\$)), \quad (43)$$

and therefore, we could obtain the same (x', w') from (x_{1-b}, w_{1-b}). Thus, the only difference in the output of hiding (x_b, w_b) with $(x_\$, w_\$)$ and hiding (x_{1-b}, w_{1-b}) with $(x'_\$, w'_\$)$ is the certificate. Further, for (x', w'), the set of elements $(x_\$, w_\$) \in \mathcal{R}'$ such that hiding (x_b, w_b) with $(x_\$, w_\$)$ results in (x', w') has the same size as the set of elements $(x'_\$, w'_\$) \in \mathcal{R}'$ such that hiding (x_{1-b}, w_{1-b}) with $(x'_\$, w'_\$)$ results in (x', w').

Since $\mathcal{A}_2$ does not receive c and $(x_\$, w_\$)$ is chosen randomly, $\mathcal{A}_2$ cannot distinguish between the two possible inputs. Hence, no adversary can do better than random guessing, showing hiding. □

With Theorems 4.10 and 4.12 and the results from Section 4.2, we note that a folding scheme with privacy preserving selective verification, is implied by the construction of a 2-folding scheme, and showing the additional few properties outlined in Theorem 4.12.

> Remark 4.13
> A folding scheme with privacy preserving selective verification can be obtained from the following 1-step transformation. Given a folding scheme with input N language instances, simply change the scheme into a scheme with input of length $2N$, where every second language instance is randomly sampled. Since this construction is essentially identical to Construction 4.7 with Construction 4.11 as the NP-statement hider, this construction is clearly also privacy preserving.

While the authors have not found any folding scheme where Construction 4.11 cannot be applied, we observe that in all examples we have investigated (see Section 4.4), we use the entire relation $\mathcal{R}$ for sampling random instances in order for Equation (40) to hold. Therefore, it is worth noting that it is an open question, if all languages in NP allow efficient sampling of random instances from the entire language. If they do, it implies that EXPTIME = NEXPTIME, which is not expected [SF90].

4.4 EXAMPLES

In this section, we consider concrete examples of NP-languages which already have folding schemes, and show that they also allow privacy preserving selective verification. Namely, we consider a folding scheme for Inner Product Relation of Committed Values from [RZ23] and the original folding scheme for Committed Relaxed R1CS from [KST22]. In [RZ23], they additionally construct folding schemes for Polynomial Commitment Openings [KZG10; BCL+21] and for Algebraic Vector Commitment Openings [CF13]. We do not consider either of these folding schemes, but note that the folding scheme for Algebraic Vector Commitment Openings is a reduction of an instance of Algebraic Vector Commitment Openings to an instance of Inner Product Relation of Committed Values, and hence our work also implies a folding scheme with privacy preserving selective verification for Algebraic Vector Commitment Openings.

Recently, other folding schemes have been proposed. Noticeably amongst them, [BC24] introduces LatticeFold, the first folding scheme that does not use an additively homomorphic commitment scheme based on the discrete logarithm problem, but rather Ajtai commitments which are based on the module SIS problem. Thus, LatticeFold is the first post-quantum secure folding scheme. LatticeFold is a general purpose scheme, and it supports folding of both low degree relations and high degree relations. In particular, it supports both R1CS and CCS [STW23].

4.4.1 *Inner Product Relation of Committed Values*

As a first example, we consider Inner Product Relation of Committed Values [BCC+16; RZ23], which are used in Bulletproofs [BBB+18].

For this example, we use Pedersen commitments for multiple values. Usually, a (single value) Pedersen commitment [Ped91] uses public parameters (or commitment key), G and H randomly sampled group elements from a group $\mathbb{G}$ over a field $\mathbb{F}$, and a commitment to $m \in \mathbb{F}$ using randomness $r \in \mathbb{F}$ is $c \leftarrow rG + mH$. The opening of c to m is (m, r). A non-hiding Pedersen commitment simply does not use the randomness r, and also removes G from the public parameters. A non-hiding Pedersen commitment for multiple values uses public parameters $G_1, \ldots, G_n \in \mathbb{G}$, and a commitment to $\mathbf{m} = (m_1, \ldots, m_n)$ is $c \leftarrow m_1 G_1 + m_2 G_2 + \cdots + m_n G_n$. Although this is called non-hiding, it is information-theoretically hiding for $n \geq 2$. The hiding versions of the Pedersen commitment scheme are perfectly hiding for $n \geq 1$, and all versions are computationally binding under the discrete logarithm assumption. As in [RZ23], we will work with the non-hiding version, but everything translates trivially to the hiding version.

Using our group notation from Section 4.1.4, a commitment key for a non-hiding Pedersen commitment for multiple values can be denoted as $[\mathbf{r}] \in \mathbb{G}^n$, and a commitment to $\mathbf{m} \in \mathbb{F}^n$ as $[c] := [\mathbf{r}]^\top \mathbf{m}$.

The language family of Inner Product Relation of Committed Values is parameterized by a group description gk and two Pedersen commitment keys $[\mathbf{r}], [\mathbf{s}] \in \mathbb{G}^n$. Statements in the language are two Pedersen commitments $[c], [d]$ and an element z from $\mathbb{F}$, and can be thought of as a claim of knowing two vectors $\mathbf{a}, \mathbf{b}$, satisfying that $[c]$ and $[d]$ are commitments to $\mathbf{a}$ and $\mathbf{b}$, and that the inner product of $\mathbf{a}$ and $\mathbf{b}$ is z. Thus, the language is defined as

$$\mathcal{L}_{\mathsf{gk},[\mathbf{r}],[\mathbf{s}]} = \left\{([c],[d],z) \mid \exists \mathbf{a},\mathbf{b} \in \mathbb{F}^n \text{ s.t. } [c] = [\mathbf{r}]^\top \mathbf{a}, [d] = [\mathbf{s}]^\top \mathbf{b}, z = \mathbf{a}^\top \mathbf{b}\right\}, \tag{44}$$

and the corresponding relation as $\mathcal{R}_{\mathsf{gk},[\mathbf{r}],[\mathbf{s}]}$. For $i \in \{1,2\}$ let

$$y_i = (([c_i],[d_i],z_i),(\mathbf{a}_i,\mathbf{b}_i)), \tag{45}$$

(supposedly) in $\mathcal{R}_{\mathsf{gk},[\mathbf{r}],[\mathbf{s}]}$ with the witness being $(\mathbf{a}_i, \mathbf{b}_i)$. We now describe a public coin protocol for folding y_1 and y_2.

1. The prover sends $z_{1,2} = \mathbf{a}_1^\top \mathbf{b}_2$ and $z_{2,1} = \mathbf{a}_2^\top \mathbf{b}_1$.
2. The verifier sends $\rho \leftarrow_\$ \mathbb{F}$.
3. The prover and verifier each construct a new statement $([c],[d],z)$ as
$$[c] = [c_1] + \rho[c_2] \tag{46}$$
$$[d] = [d_1] + \rho^2[d_2] \tag{47}$$
$$z = z_1 + \rho z_{2,1} + \rho^2 z_{1,2} + \rho^3 z_2, \tag{48}$$
and the prover also constructs a new witness $(\mathbf{a},\mathbf{b})$ as $\mathbf{a} = \mathbf{a}_1 + \rho\mathbf{a}_2, \mathbf{b} = \mathbf{b}_1 + \rho^2\mathbf{b}_2$.

Completeness of the protocol follows from straightforward calculation; if $y_1, y_2 \in \mathcal{R}_{\mathsf{gk},[\mathbf{r}],[\mathbf{s}]}$, then $(([c],[d],z),(\mathbf{a},\mathbf{b})) \in \mathcal{R}_{\mathsf{gk},[\mathbf{r}],[\mathbf{s}]}$. Knowledge soundness is a little more tricky, but, as outlined in [RZ23], it essentially follows from noticing that a prover who is able to open commitments of the form $[\alpha_1] + \rho[\alpha_2]$ should know openings to $[\alpha_1]$ and $[\alpha_2]$, since they are defined before the challenge ρ is given. Additionally, since $z_{1,2}$ and $z_{2,1}$ are defined before ρ is given, one could treat the relation that $\mathbf{a}^\top\mathbf{b} = z$ as a polynomial in ρ, that is, $\mathbf{a}(\rho)^\top\mathbf{b}(\rho) = z(\rho)$, and it can be shown that this implies both $\mathbf{a}_1^\top\mathbf{b}_1 = z_1$ and $\mathbf{a}_2^\top\mathbf{b}_2 = z_2$.

Since this protocol is public coin, the Fiat-Shamir heuristic [FS86] immediately transforms the protocol into a (non-interactive) 2-folding scheme for Inner Product Relation of Committed Values, which we describe in Construction 4.14. The folding proof π is simply the *cross terms* $z_{1,2}$ and $z_{2,1}$.

> CONSTRUCTION 4.14 *2-IPRCV.*
> Let H denote a hash function sampled from a family of cryptographic hash functions. Construct 2-IPRCV = (Fold, FoldVerify) as follows.
>
> - $\mathsf{Fold}((\mathsf{gk},[\mathbf{r}],[\mathbf{s}]),(([c_1],[d_1],z_1),(\mathbf{a}_1,\mathbf{b}_1)),(([c_2],[d_2],z_2),(\mathbf{a}_2,\mathbf{b}_2)))$
>   1. Compute the cross terms: $z_{1,2} = \mathbf{a_1}^\top\mathbf{b_2}$ and $z_{2,1} = \mathbf{a_2}^\top\mathbf{b}_1$.

2. Compute a pseudo random challenge by hashing the parameters, public inputs, and cross terms:
$$\rho = H(\mathsf{gk}, [\mathbf{r}], [\mathbf{s}], [c_1], [d_1], z_1, [c_2], [d_2], z_2, z_{1,2}, z_{2,1}). \quad (49)$$
3. Construct the folded instance using ρ:
$$[c] = [c_1] + \rho[c_2] \quad (50)$$
$$[d] = [d_1] + \rho^2[d_2] \quad (51)$$
$$z = z_1 + \rho z_{2,1} + \rho^2 z_{1,2} + \rho^3 z_2 \quad (52)$$
$$\mathbf{a} = \mathbf{a}_1 + \rho \mathbf{a}_2 \quad (53)$$
$$\mathbf{b} = \mathbf{b}_1 + \rho^2 \mathbf{b}_2, \quad (54)$$
4. Output $(([c], [d], z), (\mathbf{a}, \mathbf{b}), (z_{1,2}, z_{2,1}))$.

- $\mathsf{FoldVerify}((\mathsf{gk}, [\mathbf{r}], [\mathbf{s}]), ([c_1], [d_1], z_1), ([c_2], [d_2], z_2), ([c], [d], z), (z_{1,2}, z_{2,1}))$
1. Compute $\rho = H(\mathsf{gk}, [\mathbf{r}], [\mathbf{s}], [c_1], [d_1], z_1, [c_2], [d_2], z_2, z_{1,2}, z_{2,1})$.
2. Check that
$$[c] = [c_1] + \rho[c_2] \quad (55)$$
$$[d] = [d_1] + \rho^2[d_2] \quad (56)$$
$$z = z_1 + \rho z_{2,1} + \rho^2 z_{1,2} + \rho^3 z_2, \quad (57)$$
3. If so output 1, otherwise output 0.

The following corollary follows from the bootstrap construction in Section 4.2.1 and the extension discussed in Remark 4.4. Theorems and proofs corresponding to the construction and discussion can be found as Theorems 1 and 2 in [RZ23].

COROLLARY 4.15 IPRCV.
2-IPRCV being a 2-folding scheme for Inner Product Relation of Committed Values implies the existence of an N-folding scheme IPRCV with selective verification for Inner Product Relation of Committed Values.

In order to get a folding scheme that is also privacy preserving, we show that Construction 4.11 can be applied, and we get an NP-statement hider by folding with a randomly sampled instance from $\mathcal{R}_{\mathsf{gk},[\mathbf{r}],[\mathbf{s}]}$. For this example, we write out Construction 4.11 for Inner Product Relation of Committed Values as Construction 4.16. The proof that this construction satisfies Theorem 4.12 is handled in Theorem 4.17. Note that for this example, we sample from the entire relation, i.e., $\mathcal{R}' = \mathcal{R}_{\mathsf{gk},[\mathbf{r}],[\mathbf{s}]}$.

CONSTRUCTION 4.16 IPRCV-SH.
Let 2-IPRCV be constructed as in Construction 4.14 and $\mathsf{p} = (\mathsf{gk}, [\mathbf{r}], [\mathbf{s}])$. Construct $(\mathsf{Hide}, \mathsf{Check})$ as follows.

- $\mathsf{Hide}(\mathsf{p}, ([c_1], [d_1], z_1), (\mathbf{a}_1, \mathbf{b_1}), (([c_\$], [d_\$], z_\$), (\mathbf{a}_\$, \mathbf{b}_\$))$
1. Fold the two instances together:
$$(([c], [d], z), (\mathbf{a}, \mathbf{b}), (z_{1,2}, z_{2,1})) \leftarrow \mathsf{2\text{-}IPRCV.Fold}(\mathsf{p}, (([c_1], [d_1], z_1), (\mathbf{a}_1, \mathbf{b_1})), (([c_\$], [d_\$], z_\$), (\mathbf{a}_\$, \mathbf{b}_\$))) \quad (58)$$
2. Output $(([c], [d], z), (\mathbf{a}, \mathbf{b}), (([c_\$], [d_\$], z_\$), z_{1,2}, z_{2,1}))$

- $\mathsf{Check}(\mathsf{p}, ([c_1], [d_1], z_1), ([c], [d], z), (([c_\$], [d_\$], z_\$), z_{1,2}, z_{2,1}))$
  1. Output the result of
  $$\mathsf{2\text{-}IPRCV.FoldVerify}(\mathsf{p}, ([c_1], [d_1], z_1), ([c_\$], [d_\$], z_\$), ([c], [d], z), (z_{1,2}, z_{2,1})). \tag{59}$$

THEOREM 4.17
If 2-IPRCV is a 2-folding scheme for Inner Product Relation of Committed Values, then IPRCV-SH is an NP-statement hider for Inner Product Relation of Committed Values, in the sense of Definition 4.6, in the random oracle model.

Proof. It is sufficient to show that IPRCV-SH satisfies Theorem 4.12. By assumption 2-IPRCV satisfies Definition 4.2.

To see that $\mathcal{R}_{\mathsf{gk},[\mathbf{r}],[\mathbf{s}]}$ supports efficient sampling, observe that we can sample a random instance in $\mathcal{R}_{\mathsf{gk},[\mathbf{r}],[\mathbf{s}]}$ by sampling two vectors in $\mathbb{F}^n$, and constructing the rest of the random instance from these, as follows:

$$\begin{aligned} \mathbf{a}_\$, \mathbf{b}_\$ &\leftarrow_\$ \mathbb{F}^n & [d_\$] &= [\mathbf{s}]^\top \mathbf{b}_\$ \\ [c_\$] &= [\mathbf{r}]^\top \mathbf{a}_\$ & z_\$ &= \mathbf{a}_\$^\top \mathbf{b}_\$. \end{aligned} \tag{60}$$

We need to show for any three instances $(x_0, w_0), (x_1, w_1)$, and $(x_\$, x_\$)$, there exists an instance $(x'_\$, w'_\$)$ such that:

$$\mathsf{2\text{-}IPRCV.Fold}(\mathsf{p}, (x_0, w_0), (x_\$, w_\$)) = \mathsf{2\text{-}IPRCV.Fold}(\mathsf{p}, (x_1, w_1), (x'_\$, w'_\$)), \tag{61}$$

where we abuse notation by ignoring the folding proof. Denoting the output from folding (x_0, w_0) and $(x_\$, w_\$)$ by (x', w') and letting $\alpha \in \{0, 1, \$\}$, we use the following notation for the instances in consideration.

$$(x_\alpha, w_\alpha) = (([c_\alpha], [d_\alpha], z_\alpha), (\mathbf{a}_\alpha, \mathbf{b}_\alpha)) \tag{62}$$
$$(x'_\$, w'_\$) = (([c'_\$], [d'_\$], z'_\$), (\mathbf{a}'_\$, \mathbf{b}'_\$)). \tag{63}$$
$$(x', w') = (([c], [d], z), (\mathbf{a}, \mathbf{b})). \tag{64}$$

We prove the existence of $(x'_\$, w'_\$)$ in the random oracle model, where we replace the hash function H, used to find ρ, with a random oracle. By doing so, we can use fixed random values ρ and ξ in the folds, and assume they do not depend on the instances folded. Since the adversary will learn neither $x_\$$ nor $x'_\$$, one of which is part of the input to H, it is justified to model H as a random oracle.

When (x', w') is obtained by folding (x_0, w_0) and $(x_\$, w_\$)$, it has the following form:

$$\mathbf{a} = \mathbf{a}_0 + \rho \mathbf{a}_\$ \tag{65}$$
$$\mathbf{b} = \mathbf{b}_0 + \rho^2 \mathbf{b}_\$ \tag{66}$$
$$[c] = [c_0] + \rho[c_\$] = [c_0] + \rho[\mathbf{r}]^\top \mathbf{a}_\$ \tag{67}$$
$$[d] = [d_0] + \rho^2[d_\$] = [d_0] + \rho^2[\mathbf{s}]^\top \mathbf{b}_2 \tag{68}$$

$$\begin{aligned} z &= z_0 + \rho z_{\$,0} + \rho^2 z_{0,\$} + \rho^3 z_\$ \\ &= \mathbf{a}_0^\top \mathbf{b}_0 + \rho \mathbf{a}_\$^\top \mathbf{b}_0 + \rho^2 \mathbf{a}_0^\top \mathbf{b}_\$ + \rho^3 \mathbf{a}_\$^\top \mathbf{b}_\$. \end{aligned} \tag{69}$$

In Equations (67) to (69), we substitute in how $x_\$ = ([c_\$], [d_\$], z_\$)$ is derived from $w_\$ = (\mathbf{a}_\$, \mathbf{b}_\$)$, i.e., Equation (60), and the definition of the cross terms.

Since the goal is to have (x_1, w_1) and $(x'_\$, w'_\$)$ fold to $(([c], [d], z), (\mathbf{a}, \mathbf{b}))$, it follows from Equations (65) and (66) that we need

$$\mathbf{a}_1 + \xi \mathbf{a}'_\$ = \mathbf{a} = \mathbf{a}_0 + \rho \mathbf{a}_\$ \tag{70}$$

$$\mathbf{b}_1 + \xi^2 \mathbf{b}'_\$ = \mathbf{b} = \mathbf{b}_0 + \rho^2 \mathbf{b}_\$, \tag{71}$$

and therefore we fix $\mathbf{a}'_\$$ and $\mathbf{b}'_\$$ such that

$$\xi \mathbf{a}'_\$ = \mathbf{a}_0 + \rho \mathbf{a}_\$ - \mathbf{a}_1 \tag{72}$$

$$\xi^2 \mathbf{b}'_\$ = \mathbf{b}_0 + \rho^2 \mathbf{b}_\$ - \mathbf{b}_1. \tag{73}$$

Deriving $x'_\$$ from the now fixed $w'_\$ = (\mathbf{a}'_\$, \mathbf{b}'_\$)$ by the same calculations as in Equation (60), we obtain the instance $(x'_\$, w'_\$) \in \mathcal{R}_{\mathsf{gk},[\mathbf{r}],[\mathbf{s}]}$.

It now suffices to verify that when (x_1, w_1) is folded with $(x'_\$, w'_\$)$ using randomness ξ, we get (x', w'). We first verify $[c]$:

$$[c_1] + \xi [c'_\$] = [c_1] + \xi [\mathbf{r}]^\top \mathbf{a}'_\$ = [c_1] + [\mathbf{r}]^\top (\xi \mathbf{a}'_\$) \tag{74}$$

$$= [c_1] + [\mathbf{r}]^\top (\mathbf{a}_0 + \rho \mathbf{a}_\$ - \mathbf{a}_1) \tag{75}$$

$$= [c_1] + \underbrace{[\mathbf{r}]^\top \mathbf{a}_0}_{=[c_0]} + \rho \underbrace{[\mathbf{r}]^\top \mathbf{a}_\$}_{=[c_\$]} - \underbrace{[\mathbf{r}]^\top \mathbf{a}_1}_{=[c_1]} \tag{76}$$

$$= [c_0] + \rho [c_\$] = [c]. \tag{77}$$

A very similar calculation can be done to verify $[d]$:

$$[d_1] + \xi^2 [d_\$] = [d_1] + [\mathbf{s}]^\top \xi^2 \mathbf{b}'_\$ \tag{78}$$

$$= [d_1] + [\mathbf{s}]^\top (\mathbf{b}_0 + \rho^2 \mathbf{b}_\$ - \mathbf{b}_1) \tag{79}$$

$$= [d_1] + [d_0] + \rho^2 [d_\$] - [d_1] \tag{80}$$

$$= [d_0] + \rho^2 [d_\$] = [d]. \tag{81}$$

Verifying z is done by showing that

$$z = z_1 + \xi z_{\$,1} + \xi^2 z_{1,\$} + \xi^3 z'_\$. \tag{82}$$

As a first step, we expand the four terms in Equation (82) separately.

$$z_1 = \mathbf{a}_1^\top \mathbf{b}_1 \tag{83}$$

$$\begin{aligned} \xi z_{\$,1} &= \xi (\mathbf{a}'_\$)^\top \mathbf{b}_1 = (\mathbf{a}_0 + \rho \mathbf{a}_\$ - \mathbf{a}_1)^\top \mathbf{b}_1 \\ &= \mathbf{a}_0^\top \mathbf{b}_1 + \rho \mathbf{a}_\$^\top \mathbf{b}_1 - \mathbf{a}_1^\top \mathbf{b}_1 \end{aligned} \tag{84}$$

$$\begin{aligned} \xi^2 z_{1,\$} &= \mathbf{a}_1^\top (\xi^2 \mathbf{b}'_\$) = \mathbf{a}_1^\top (\mathbf{b}_0 + \rho^2 \mathbf{b}_\$ - \mathbf{b}_1) \\ &= \mathbf{a}_1^\top \mathbf{b}_0 + \rho^2 \mathbf{a}_1^\top \mathbf{b}_\$ - \mathbf{a}_1^\top \mathbf{b}_1 \end{aligned} \tag{85}$$

$$\begin{aligned}\xi^3 z'_{\$} &= (\xi \mathbf{a}'_{\$})^\top \xi^2 \mathbf{b}'_{\$} = (\mathbf{a}_0 + \rho \mathbf{a}_{\$} - \mathbf{a}_1)^\top (\mathbf{b}_0 + \rho^2 \mathbf{b}_{\$} - \mathbf{b}_1) \\ &= \mathbf{a}_0^\top (\mathbf{b}_0 + \rho^2 \mathbf{b}_{\$} - \mathbf{b}_1) + \mathbf{a}_{\$}^\top (\rho \mathbf{b}_0 + \rho^3 \mathbf{b}_{\$} - \rho \mathbf{b}_1) \\ &\quad - \mathbf{a}_1^\top (\mathbf{b}_0 + \rho^2 \mathbf{b}_{\$} - \mathbf{b}_1). \end{aligned} \tag{86}$$

Inserting Equations (83) to (86) into Equation (82) and factoring out the $\mathbf{a}$ terms, yields

$$\begin{aligned}(82) &= \mathbf{a}_1^\top (\mathbf{b}_1 - \mathbf{b}_1 + \mathbf{b}_0 + \rho^2 \mathbf{b}_{\$} - \mathbf{b}_1 - \mathbf{b}_0 - \rho^2 \mathbf{b}_{\$} + \mathbf{b}_1) \\ &\quad + (\mathbf{a}_0)^\top (\mathbf{b}_1 + \mathbf{b}_0 + \rho^2 \mathbf{b}_{\$} - \mathbf{b}_1) \qquad (87) \\ &\quad + \mathbf{a}_{\$}^\top (\rho \mathbf{b}_1 + \rho \mathbf{b}_0 + \rho^3 \mathbf{b}_{\$} - \rho \mathbf{b}_1) \\ &= \mathbf{a}_1^\top \mathbf{0} + \mathbf{a}_0^\top (\mathbf{b}_0 + \rho^2 \mathbf{b}_{\$}) + \mathbf{a}_{\$}^\top (\rho \mathbf{b}_0 + \rho^3 \mathbf{b}_{\$}) \qquad (88) \\ &= \mathbf{a}_0^\top \mathbf{b}_0 + \rho^2 (\mathbf{a}_0)^\top \mathbf{b}_{\$} + \rho \mathbf{a}_{\$}^\top \mathbf{b}_0 + \rho^3 \mathbf{a}_{\$}^\top \mathbf{b}_{\$} \qquad (89) \\ &= z_0 + \rho^2 z_{0,\$} + \rho z_{\$,0} + \rho^3 z_{\$} = z. \qquad (90)\end{aligned}$$

We have now shown that there is a $(x'_{\$}, w'_{\$})$ such that Equation (61) holds in the random oracle model.

Finally, we observe that for a fixed (x', w'), it follows from Equations (70) and (71) that there is exactly one instance $(x_{\$}, w_{\$})$ folding (x_0, w_0) into (x', w') for each non-zero randomness ρ, and equivalently for each ξ, there is one $(x'_{\$}, w'_{\$})$ folding (x_1, w_1) into (x', w'). Hence, IPRCV-SH satisfies Theorem 4.12. □

> Corollary 4.18
> There is a folding scheme with privacy preserving selective verification for Inner Product Relation of Committed Values in the random oracle model.

4.4.2 *Committed Relaxed R1CS*

Committed Relaxed R1CS is the language used in the original paper introducing folding schemes [KST22]. The language is a folding amenable generalization of Rank One Constraint Systems (R1CS) [SBV+13; GGPR13], and a classical language used for many proof systems [Gro16; GWC19; BCR+19; KS22]. R1CS is a satisfiability flavored characterization of the complexity class NP. Roughly, R1CS works as follows. For the three parameters, $m \times m$ matrices $A, B, C \in \mathbb{F}^{m \times m}$, an instance of R1CS is $\mathbf{x} \in \mathbb{F}^n$ with $n < m$ for which there is a witness $\mathbf{w} \in \mathbb{F}^{m-n-1}$ such that with $\mathbf{z} = (\mathbf{w}, \mathbf{x}, 1)^\top$ we have

$$A\mathbf{z} \circ B\mathbf{z} = C\mathbf{z}, \tag{91}$$

where $\circ$ denotes entry wise multiplication, also called the Hadamard product.

To make R1CS amenable to folding, the structure is modified to have $u \in \mathbb{F}$, rather than 1, as the last entry in $\mathbf{z}$, and as a scalar in front of $C\mathbf{z}$. Additionally, an error term $\mathbf{e}$ is introduced. To keep the protocol zero-knowledge, commitments for $\mathbf{w}$ and $\mathbf{e}$ are introduced, using an additively

homomorphic commitment scheme, for example Pedersen commitments. For notation, we write $\overline{x} \leftarrow \mathsf{Com}(x, r_x)$, meaning that $\overline{x}$ is a commitment to x using randomness r_x. With this notation, the language Committed Relaxed R1CS is

$$\mathcal{L}_{A,B,C} = \left\{ (u, \mathbf{x}, \overline{\mathbf{e}}, \overline{\mathbf{w}}) \middle| \exists (\mathbf{e}, r_{\mathbf{e}}, \mathbf{w}, r_{\mathbf{w}}) : \begin{array}{c} \mathbf{z} := (\mathbf{w}, \mathbf{x}, u) \\ A\mathbf{z} \circ B\mathbf{z} = uC\mathbf{z} + \mathbf{e} \\ \overline{\mathbf{e}} \leftarrow \mathsf{Com}(\mathbf{e}, r_{\mathbf{e}}) \\ \overline{\mathbf{w}} \leftarrow \mathsf{Com}(\mathbf{w}, r_{\mathbf{w}}) \end{array} \right\}, \tag{92}$$

with a corresponding relation $\mathcal{R}_{A,B,C}$. Note that since R1CS is NP-complete and included in Committed Relaxed R1CS, Committed Relaxed R1CS captures NP. In [KST22], they claim that Committed Relaxed R1CS is NP-complete, because it "contains R1CS", but this depends on the commitment schemes used. In particular, when mapping an instance of R1CS to an instance of Committed Relaxed R1CS, the mapping needs to produce a commitment to the witness, without knowing the witness. Thus, a commitment scheme that is not information-theoretically binding must be used for the witness (this could, for example, be the information-theoretically hiding Pedersen commitments discussed in Section 4.4.1). However, for "no"-instances of R1CS, the commitment to the error vector being 0 must be information-theoretically binding, or else the instance becomes a "yes"-instance of Committed Relaxed R1CS, just by using a suitable error vector. Thus, for Committed Relaxed R1CS to be NP-complete because it "contains R1CS", one must be be careful in using the right commitment schemes.

Following [KST22], a public coin protocol for folding two instances of Committed Relaxed R1CS can be constructed as follows. For $i \in \{0, 1\}$, denote the two instances as

$$y_i = ((u_i, \mathbf{x}_i, \overline{\mathbf{e}_i}, \overline{\mathbf{w}_i}), (\mathbf{e}_i, r_{\mathbf{e}_i}, \mathbf{w}_i, r_{\mathbf{w}_i})) \in \mathcal{R}_{A,B,C}. \tag{93}$$

1. The prover sends $\overline{t} \leftarrow \mathsf{Com}(t, r_t)$ where $r_t \leftarrow_\$ \mathbb{F}$ and
$$t = A\mathbf{z}_1 \circ B\mathbf{z}_2 + A\mathbf{z}_2 \circ B\mathbf{z}_1 - u_1 C\mathbf{z}_2 - u_2 C\mathbf{z}_1. \tag{94}$$
2. The verifier sends $\rho \leftarrow_\$ \mathbb{F}$.
3. Both the prover and verifier construct the folded instance $(u, \mathbf{x}, \overline{\mathbf{e}}, \overline{\mathbf{w}})$ where[3]
$$\begin{aligned} u &= u_1 + \rho u_2 & \overline{\mathbf{e}} &= \overline{\mathbf{e}_1} + \rho \overline{t} + \rho^2 \overline{\mathbf{e}_2} \\ \mathbf{x} &= \mathbf{x}_1 + \rho \mathbf{x}_2 & \overline{\mathbf{w}} &= \overline{\mathbf{w}_1} + \rho \overline{\mathbf{w}_2}. \end{aligned} \tag{95}$$
Additionally, the prover constructs a witness $(\mathbf{e}, r_{\mathbf{e}}, \mathbf{w}, r_{\mathbf{w}})$ for the folded instance, where
$$\begin{aligned} \mathbf{e} &= \mathbf{e}_1 + \rho t + \rho^2 \mathbf{e}_2 & \mathbf{w} &= \mathbf{w}_1 + \rho \mathbf{w}_2 \\ r_{\mathbf{e}} &= r_{\mathbf{e}_1} + \rho r_t + \rho^2 r_{\mathbf{e}_2} & r_{\mathbf{w}} &= r_{\mathbf{w}_1} + \rho r_{\mathbf{w}_2}. \end{aligned} \tag{96}$$

This protocol can be turned into a (non-interactive) 2-folding scheme using the Fiat-Shamir heuristic, and the 2-folding scheme can be bootstrapped using the techniques mentioned in Section 4.2.1 to an N-folding

3 Recall that the verifier knows the values committed to by $\overline{\mathbf{e}}$ and $\overline{\mathbf{w}}$, and Pedersen commitments allow noninteractively multiplying commitments by scalars and adding commitments.

scheme with selective verification. Denote this folding scheme CR-R1CS. The security of CR-R1CS follows from the same type of arguments as the scheme in Section 4.4.1, and a proof can be found in [KST22]. It follows from the proof of the bootstrap construction in [RZ23] that CR-R1CS is selectively verifiable.

In order to get a folding scheme with privacy preserving selective verification for Committed Relaxed R1CS, we first instantiate Construction 4.11 with CR-R1CS to get a statement hider SH-CR-R1CS. Similar to the IPRCV statement hider, we use the entire relation, $\mathcal{R}_{A,B,C}$, as the sample space. We can then instantiate Construction 4.7 with CR-R1CS and SH-CR-R1CS to get a folding scheme with privacy preserving selective verification for Committed Relaxed R1CS. We denote this instantiation of Construction 4.7 as PP-CR-R1CS. We now show that PP-CR-R1CS is a folding scheme with privacy preserving selective verification in the random oracle model.

> THEOREM 4.19
> Assuming that CR-R1CS is a folding scheme with selective verification, PP-CR-R1CS, constructed as described in the previous paragraph, is a folding scheme with privacy preserving selective verification for Committed Relaxed R1CS, in the random oracle model.

Proof. From Theorems 4.10 and 4.12, it follows that in order to show that PP-CR-R1CS satisfies Definitions 4.2, 4.3 and 4.5, it is sufficient to show that CR-R1CS satisfies Definition 4.3 and that Theorem 4.12 applies. By assumption, CR-R1CS satisfies Definitions 4.2 and 4.3.

Efficient sampling from the entire relation space can be obtained as follows: First, sample random vectors $\mathbf{x} \in \mathbb{F}^n, \mathbf{w} \in \mathbb{F}^{m-n-1}$ and $u \in \mathbb{F}$. Then, with $\mathbf{z} = (\mathbf{w}, \mathbf{x}, u)^\top$, set

$$\mathbf{e} := A\mathbf{z} \circ B\mathbf{z} - uC\mathbf{z}, \tag{97}$$

and generate commitments to $\mathbf{w}$ and $\mathbf{e}$ with randomness $r_{\mathbf{w}}, r_{\mathbf{e}} \leftarrow_{\$} \mathbb{F}$:

$$\overline{\mathbf{e}} \leftarrow \mathsf{Com}(\mathbf{e}, r_{\mathbf{e}}) \qquad \overline{\mathbf{w}} \leftarrow \mathsf{Com}(\mathbf{w}, r_{\mathbf{w}}). \tag{98}$$

The random instance is now given by $((u, \mathbf{x}, \overline{\mathbf{e}}, \overline{\mathbf{w}}), (\mathbf{e}, r_{\mathbf{e}}, w, r_{\mathbf{w}}))$. It follows from Equation (97) that the instance by definition is in $\mathcal{R}_{A,B,C}$, and since $\mathbf{z}$ is a random vector in $\mathbb{F}^m$, the instance is chosen randomly from the entire space.

The next criterion we show is that for any two instances, y_1 and y_1', with

$$y_1 = ((u_1, \mathbf{x}_1, \overline{\mathbf{e}_1}, \overline{\mathbf{w}_1}), (\mathbf{e}_1, r_{\mathbf{e}_1}, \mathbf{w}_1, r_{\mathbf{w}_1})) \tag{99}$$

$$y_1' = ((u_1', \mathbf{x}_1', \overline{\mathbf{e}_1'}, \overline{\mathbf{w}_1'}), (\mathbf{e}_1', r_{\mathbf{e}_1'}, \mathbf{w}_1', r_{\mathbf{w}_1'})), \tag{100}$$

and third instance $y_2 = ((u_2, \mathbf{x}_2, \overline{\mathbf{e}_2}, \overline{\mathbf{w}_2}), (\mathbf{e}_2, r_{\mathbf{e}_2}, \mathbf{w}_2, r_{\mathbf{w}_2}))$, there is an instance $y_2' = ((u_2', \mathbf{x}_2', \overline{\mathbf{e}_i'}, \overline{\mathbf{w}_2'}), (\mathbf{e}_2', r_{\mathbf{e}_2'}, \mathbf{w}_2', r_{\mathbf{w}_2'}))$, such that the statement and witness obtained by folding y_1 and y_2 is the same as the statement and

witness obtained by folding y_1' and y_2'. That is, abusing notation by ignoring the proof of folding, we find y_2' such that

$$\mathsf{Fold}(y_1, y_2) = \mathsf{Fold}(y_1', y_2'). \tag{101}$$

We show this in the random oracle model, and denote the randomness used for the first fold as ρ and for the second fold as ξ. To satisfy Equation (101), the following equations must hold:

$$\mathbf{x}_1 + \rho \mathbf{x}_2 = \mathbf{x}_1' + \xi \mathbf{x}_2' \tag{102}$$
$$u_1 + \rho u_2 = u_1' + \xi u_2' \tag{103}$$
$$\mathbf{w}_1 + \rho \mathbf{w}_2 = \mathbf{w}_1' + \xi \mathbf{w}_2' \tag{104}$$
$$\mathbf{e}_1 + \rho \cdot t + \rho^2 \mathbf{e}_2 = \mathbf{e}_1' + \xi t' + \xi^2 \mathbf{e}_2', \tag{105}$$

where t and t' are the cross terms from Equation (94), corresponding to $\mathsf{Fold}(y_1, y_2)$ and $\mathsf{Fold}(y_1', y_2')$, respectively. Isolating the terms from y_2', gives us

$$\mathbf{x}_2' = \rho^{-1}(\mathbf{x}_1 + \rho \mathbf{x}_2 - \mathbf{x}_1') \tag{106}$$
$$u_2' = \rho^{-1}(u_1 + \rho u_2 - u_1') \tag{107}$$
$$\mathbf{w}_2' = \rho^{-1}(\mathbf{w}_1 + \rho \mathbf{w}_2 - \mathbf{w}_1') \tag{108}$$
$$\mathbf{e}_2' = \rho^{-2}(\mathbf{e}_1 + \rho t + \rho^2 \mathbf{e}_2 - \mathbf{e}_1' - \rho t'). \tag{109}$$

Constructing the commitments to $\mathbf{w}_2'$ and $\mathbf{e}_2'$ (and their randomness) from their respective parts, we are left with an instance

$$y_2' = ((u_2', \mathbf{x}_2', \overline{\mathbf{e}_2'}, \overline{\mathbf{w}_2'}), (\mathbf{e}_2', r_{\mathbf{e}_2'}, \mathbf{w}_2', r_{\mathbf{w}_2'})) \tag{110}$$

$$\begin{aligned} &= ((\xi^{-1}(u_1 + \rho u_2 - u_1'), \xi^{-1}(\mathbf{x}_1 + \rho \mathbf{x}_2 - \mathbf{x}_1'), \\ &\quad \xi^{-2}(\overline{\mathbf{e}_1} + \rho \overline{t} + \rho^2 \overline{\mathbf{e}_2} - \overline{\mathbf{e}_1'} - \xi \overline{t'}), \xi^{-1}(\overline{\mathbf{w}_1} + \rho \overline{\mathbf{w}_2} - \overline{\mathbf{w}_1'})), \\ &\quad (\xi^{-2}(\mathbf{e}_1 + \rho t + \rho^2 \mathbf{e}_2 - \mathbf{e}_1' - \xi t'), \xi^{-2}(r_{\mathbf{e}_1} + \rho r_t + \rho^2 r_{\mathbf{e}_2} - r_{\mathbf{e}_1'} - \xi r_{t'}), \\ &\quad \xi^{-1}(\mathbf{w}_1 + \rho \mathbf{w}_2 - \mathbf{w}_1'), \xi^{-1}(r_{\mathbf{w}_1} + \rho r_{\mathbf{w}_2} - r_{\mathbf{w}_1'}))), \end{aligned} \tag{111}$$

which by construction satisfies Equation (101). We need to verify that the instance is indeed in $\mathcal{R}_{A,B,C}$. By inspection, the commitments are correct, so it suffices to verify that

$$A\mathbf{z}_2' \circ B\mathbf{z}_2' = u_2' C\mathbf{z}_2' + \mathbf{e}_2'. \tag{112}$$

By construction

$$\mathbf{z}_2' := \begin{pmatrix} \mathbf{w}_2' \\ \mathbf{x}_2' \\ u_2' \end{pmatrix} = \begin{pmatrix} \xi^{-1}(\mathbf{w}_1 + \rho \mathbf{w}_2 - \mathbf{w}_1') \\ \xi^{-1}(\mathbf{x}_1 + \rho \mathbf{x}_2 - \mathbf{x}_1') \\ \xi^{-1}(u_1 + \rho u_2 - u_1') \end{pmatrix} = \xi^{-1}(\mathbf{z}_1 + \rho \mathbf{z}_2 - \mathbf{z}_1'). \tag{113}$$

We first expand the left side of Equation (112) using the distributive laws for entry-wise multiplication.

$$A\mathbf{z}_2' \circ B\mathbf{z}_2' = A(\xi^{-1}(\mathbf{z}_1 + \rho \mathbf{z}_2 - \mathbf{z}_1')) \circ B(\xi^{-1}(\mathbf{z}_1 + \rho \mathbf{z}_2 - \mathbf{z}_1')) \tag{114}$$

$$= \xi^{-2}(A\mathbf{z}_1 + \rho A\mathbf{z}_2 - A\mathbf{z}'_1) \circ (B\mathbf{z}_1 + \rho B\mathbf{z}_2 - B\mathbf{z}'_1)) \tag{115}$$

$$\begin{aligned} &= \xi^{-2}(A\mathbf{z}_1 \circ B\mathbf{z}_1 + \rho A\mathbf{z}_1 \circ B\mathbf{z}_2 - A\mathbf{z}_1 \circ B\mathbf{z}'_1 \\ &\quad + \rho A\mathbf{z}_2 \circ B\mathbf{z}_1 + \rho^2 A\mathbf{z}_2 \circ B\mathbf{z}_2 - \rho A\mathbf{z}_2 \circ B\mathbf{z}'_1 \\ &\quad - A\mathbf{z}'_1 \circ B\mathbf{z}_1 - \rho A\mathbf{z}'_1 \circ B\mathbf{z}_2 + A\mathbf{z}'_1 \circ B\mathbf{z}'_1) \end{aligned} \tag{116}$$

$$\begin{aligned} &= \rho^2\xi^{-2}(A\mathbf{z}_2 \circ B\mathbf{z}_2) \\ &\quad + \rho\xi^{-2}(A\mathbf{z}_1 \circ B\mathbf{z}_2 + A\mathbf{z}_2 \circ B\mathbf{z}_1 - A\mathbf{z}_2 \circ B\mathbf{z}'_1 - A\mathbf{z}'_1 \circ B\mathbf{z}_2) \\ &\quad + \xi^{-2}(A\mathbf{z}_1 \circ B\mathbf{z}_1 - A\mathbf{z}_1 \circ B\mathbf{z}'_1 - A\mathbf{z}'_1 \circ B\mathbf{z}_1 + A\mathbf{z}'_1 \circ B\mathbf{z}'_1) \end{aligned} \tag{117}$$

$$\begin{aligned} &= \xi^{-2}(\rho^2(u_2 C\mathbf{z}_2 + \mathbf{e}_2) \\ &\quad + \rho(A\mathbf{z}_1 \circ B\mathbf{z}_2 + A\mathbf{z}_2 \circ B\mathbf{z}_1 - A\mathbf{z}_2 \circ B\mathbf{z}'_1 - A\mathbf{z}'_1 \circ B\mathbf{z}_2) \\ &\quad + u_1 C\mathbf{z}_1 + \mathbf{e}_1 - A\mathbf{z}_1 \circ B\mathbf{z}'_1 - A\mathbf{z}'_1 \circ B\mathbf{z}_1 + u'_1 C\mathbf{z}'_1 + \mathbf{e}'_1). \end{aligned} \tag{118}$$

Before we expand the right side of Equation (112), we expand the error term $\mathbf{e}'_2$. This is done by inserting the cross terms t and t', multiplying out, inserting u'_2 and $\mathbf{z}'_2$, and then multiplying out again. We obtain that

$$\mathbf{e}'_2 = \xi^{-2}(\mathbf{e}_1 + \rho t + \rho^2 \mathbf{e}_2 - \mathbf{e}'_1 - \xi t') \tag{119}$$

$$\begin{aligned} &= \xi^{-2}(\mathbf{e}_1 + \rho(A\mathbf{z}_1 \circ B\mathbf{z}_2 + A\mathbf{z}_2 \circ B\mathbf{z}_1 - u_1 C\mathbf{z}_2 - u_2 C\mathbf{z}_1) + \rho^2 \mathbf{e}_2 \\ &\quad - \mathbf{e}'_1 - \xi(A\mathbf{z}'_1 \circ B\mathbf{z}'_2 + A\mathbf{z}'_2 \circ B\mathbf{z}'_1 - u'_1 C\mathbf{z}'_2 - u'_2 C\mathbf{z}'_1)) \end{aligned} \tag{120}$$

$$\begin{aligned} &= \xi^{-2}(\mathbf{e}_1 + \rho(A\mathbf{z}_1 \circ B\mathbf{z}_2 + A\mathbf{z}_2 \circ B\mathbf{z}_1 - u_1 C\mathbf{z}_2 - u_2 C\mathbf{z}_1) \\ &\quad + \rho^2 \mathbf{e}_2 - \mathbf{e}'_1 \\ &\quad - \xi(A\mathbf{z}'_1 \circ B(\xi^{-1}(\mathbf{z}_1 + \rho\mathbf{z}_2 - \mathbf{z}'_1)) \\ &\quad\quad + A(\xi^{-1}(\mathbf{z}_1 + \rho\mathbf{z}_2 - \mathbf{z}'_1)) \circ B\mathbf{z}'_1 \\ &\quad\quad - u'_1 C(\xi^{-1}(\mathbf{z}_1 + \rho\mathbf{z}_2 - \mathbf{z}'_1)) \\ &\quad\quad - \xi^{-1}(u_1 + \rho u_2 - u'_1) C\mathbf{z}'_1)) \end{aligned} \tag{121}$$

$$\begin{aligned} &= \xi^{-2}(\mathbf{e}_1 + \rho(A\mathbf{z}_1 \circ B\mathbf{z}_2 + A\mathbf{z}_2 \circ B\mathbf{z}_1 - u_1 C\mathbf{z}_2 - u_2 C\mathbf{z}_1) \\ &\quad + \rho^2 \mathbf{e}_2 - \mathbf{e}'_1 - A\mathbf{z}'_1 \circ B\mathbf{z}_1 - \rho A\mathbf{z}'_1 \circ B\mathbf{z}_2 + A\mathbf{z}'_1 \circ B\mathbf{z}'_1 \\ &\quad - A\mathbf{z}_1 \circ B\mathbf{z}'_1 - \rho A\mathbf{z}_2 \circ B\mathbf{z}'_1 + A\mathbf{z}'_1 \circ B\mathbf{z}'_1 + u'_1 C\mathbf{z}_1 \\ &\quad + \rho u'_1 C\mathbf{z}_2 - u'_1 C\mathbf{z}'_1 + u_1 C\mathbf{z}'_1 + \rho u_2 C\mathbf{z}'_1 - u'_1 C\mathbf{z}'_1) \end{aligned} \tag{122}$$

$$\begin{aligned} &= \rho^2\xi^{-2}\mathbf{e}_2 + \rho\xi^{-2}(A\mathbf{z}_1 \circ B\mathbf{z}_2 + A\mathbf{z}_2 \circ B\mathbf{z}_1 - u_1 C\mathbf{z}_2 - u_2 C\mathbf{z}_1 \\ &\quad\quad - A\mathbf{z}'_1 \circ B\mathbf{z}_2 - A\mathbf{z}_2 \circ B\mathbf{z}'_1 + u'_1 C\mathbf{z}_2 + u_2 C\mathbf{z}'_1) \\ &\quad + \xi^{-2}(\mathbf{e}_1 - \mathbf{e}'_1 - A\mathbf{z}'_1 \circ B\mathbf{z}_1 + A\mathbf{z}'_1 \circ B\mathbf{z}'_1 - A\mathbf{z}_1 \circ B\mathbf{z}'_1 \\ &\quad\quad + A\mathbf{z}'_1 \circ B\mathbf{z}'_1 + u'_1 C\mathbf{z}_1 - u'_1 C\mathbf{z}'_1 + u_1 C\mathbf{z}'_1 - u'_1 C\mathbf{z}'_1). \end{aligned} \tag{123}$$

$$\begin{aligned} &= \rho^2\xi^{-2}\mathbf{e}_2 + \rho\xi^{-2}(A\mathbf{z}_1 \circ B\mathbf{z}_2 + A\mathbf{z}_2 \circ B\mathbf{z}_1 - u_1 C\mathbf{z}_2 - u_2 C\mathbf{z}_1 \\ &\quad\quad - A\mathbf{z}'_1 \circ B\mathbf{z}_2 - A\mathbf{z}_2 \circ B\mathbf{z}'_1 + u'_1 C\mathbf{z}_2 + u_2 C\mathbf{z}'_1) \\ &\quad + \xi^{-2}(\mathbf{e}_1 + \mathbf{e}'_1 - A\mathbf{z}'_1 \circ B\mathbf{z}_1 - A\mathbf{z}_1 \circ B\mathbf{z}'_1 + u'_1 C\mathbf{z}_1 + u_1 C\mathbf{z}'_1). \end{aligned} \tag{124}$$

For Equation (124), we applied that $A\mathbf{z}'_1 \circ B\mathbf{z}'_1 = u'_1 C\mathbf{z}'_1 + \mathbf{e}'_1$ twice. We are now ready to expand the right side of Equation (112). At Equation (128), we insert Equation (124) in place of $\mathbf{e}'_2$, and cancel out where applicable.

$$u'_2 C\mathbf{z}'_2 + \mathbf{e}'_2 = \xi^{-1}(u_1 + \rho u_2 - u'_1)C(\xi^{-1}(\mathbf{z}_1 + \rho\mathbf{z}_2 - \mathbf{z}'_1)) + \mathbf{e}'_2 \tag{125}$$

$$\begin{aligned} &= \xi^{-2}(u_1 C\mathbf{z}_1 + \rho u_1 C\mathbf{z}_2 - u_1 C\mathbf{z}'_1 + \rho u_2 C\mathbf{z}_1 \\ &\qquad + \rho^2 u_2 C\mathbf{z}_2 - \rho u_2 C\mathbf{z}'_1 \\ &\qquad - u'_1 C\mathbf{z}_1 - \rho u'_1 C\mathbf{z}_2 + u'_1 C\mathbf{z}'_1) + \mathbf{e}'_2 \end{aligned} \tag{126}$$

$$\begin{aligned} &= \rho^2\xi^{-2}(u_2 C\mathbf{z}_2) + \rho\xi^{-2}(u_1 C\mathbf{z}_2 + u_2 C\mathbf{z}_1 \\ &\qquad - u_2 C\mathbf{z}'_1 - u'_1 C\mathbf{z}_2) \\ &\quad + \xi^{-2}(u_1 C\mathbf{z}_1 - u_1 C\mathbf{z}'_1 - u'_1 C\mathbf{z}_1 + u'_1 C\mathbf{z}'_1) + \mathbf{e}'_2 \end{aligned} \tag{127}$$

$$\begin{aligned} &= \rho^2\xi^{-2}(u_2 C\mathbf{z}_2 + \mathbf{e}_2) + \rho\xi^{-2}(A\mathbf{z}_1 \circ B\mathbf{z}_2 + A\mathbf{z}_2 \circ B\mathbf{z}_1 \\ &\qquad - A\mathbf{z}'_1 \circ B\mathbf{z}_2 - A\mathbf{z}_2 \circ B\mathbf{z}'_1) \\ &\quad + \xi^{-2}(u_1 C\mathbf{z}_1 + \mathbf{e}_1 + u'_1 C\mathbf{z}'_1 + \\ &\qquad \mathbf{e}'_1 - A\mathbf{z}'_1 \circ B\mathbf{z}_1 - A\mathbf{z}_1 \circ B\mathbf{z}'_1). \end{aligned} \tag{128}$$

Equation (112) can now be verified, simply by comparing Equations (118) and (128). Thus, we have shown the existence of $y'_2 \in \mathcal{R}_{A,B,C}$ such that Equation (101) holds.

Finally, it can be observed from Equations (102) to (105) that each unique pair ρ and y_2 hiding y_1 as a fixed instance corresponds to a unique pair ξ and y'_2 hiding y'_1 as the same instance, showing that the last criterion of Theorem 4.12 is satisfied, and hence finishing the proof of Theorem 4.19. □

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for helpful feedback that improved the quality of this article.

A

ADDING QUOTABLE SIGNATURES TO THE TRANSPARENCY REPERTOIRE IN DATA JOURNALISM

Marília Gehrke and Simon Erfurth. Adding quotable signatures to the transparency repertoire in data journalism. In *Joint Computation+Journalism Symposium and European Data & Computational Journalism Conference 2023*, 2023. URL: https://www.datajconf.com/papers/CJ_DataJConf_2023_paper_17.pdf

ABSTRACT Fabricated content falsely attributed to reputable news sources is one of the significant challenges for journalism today. One of the manipulation methods is to copy the layout of news websites and substitute the original text. The manipulated version is then recirculated, making it hard to assess the reliability and trace the origin of such "information." Offering an exploratory, descriptive, and solution-oriented approach, we present examples of how this manipulation threatens news outlets and can escalate to data journalism and other specialized forms of news reporting. One reason for that is people's overreliance on numbers and data visualizations as cues to assess the trustworthiness of the content. Then, we suggest that news organizations and social media platforms incorporate a tool to make the digital information environment safer for users and readers. By presenting quotable signature schemes, a cryptography-based solution, we claim that the transparency repertoire in journalism can be improved and extended.

A.1 INTRODUCTION

In this paper, we argue that using quotable signature schemes can enhance and extend the repertoire of transparency strategies in data journalism to prevent or combat mis- and disinformation spread. Our approach here is descriptive, exploratory, and solution-based. We start the study with a theoretical background, present examples of how manipulation impersonates reputable news brands, and offer a solution based on a cryptographic primitive called quotable signature schemes. Lastly, we offer a prototype to trace a publication's provenance when excerpts from it are shared on social media.

Our research is in line with literature that debates fabricated information that often mimics the format of news [LBB+18], relies on numbers as cues to manipulate readers [GB21; PLS23], and even recirculates journalistic products by taking them out of their original context [SR21]. Thus, our goal is to describe how disinformation spread is currently problematic to news organizations and has the potential to escalate to data jour-

nalism websites in the future, taking into account that numbers and data visualizations are potent cues to perceived credibility [PLS23].

This study is part of the *Trust and News Authenticity* interdisciplinary project, connected with the Digital Democracy Centre (DDC) at the University of Southern Denmark (SDU). All in all, we suggest that data journalists use computational tools to make the information environment safer and more transparent for users and readers. As part of transparency strategies, quotable signature schemes can be extended to news sources inside the journalistic articles, which allows one to authenticate information's provenance. In the future, the mechanism can even be extended to multimedia forms, such as pictures and videos.

A.2 THEORETICAL BACKGROUND

A.2.1 *Transparency in data journalism practices*

Data journalism has ascended since the late 2000s in Europe and the United States, mainly due to the advance of Freedom of Information Access legislation [Cod15; Rog13]. This data-driven journalism practice encompasses investigations that primarily rely on public databases, even though leaked documents can also be used as sources.

Since its theoretical roots trace back to Precision Journalism [Mey02], which aimed to posit journalism closer to the scientific method, data journalism investigations might start with a hypothesis to be tested, followed by data analysis, visualization, and communication of the reporting method – that is, methodological transparency. One of the most common requirements of openness is focused on the replicability and/or reproducibility of the analysis, allowing the audience to verify information and find the same results as the journalists [Geh22; GM17; Mey02].

In summary, transparency means disclosing reporting practices and being clear about the origin of news sources and the methodology adopted. Almost a decade ago, transparency in digital journalism, which includes data-driven approaches, was seen by scholars as a way to establish credibility and reduce mistrust among audiences [Cod15; Kar10]. The optimism was mainly connected with the Web allowing the use of hypertext and, therefore, new layers of information. Recently, though, only a couple of investigations presented evidence that transparency could increase perceived credibility [JS21], and some scholars argue that trust is a prerequisite for openness to be effective [Kar22].

Despite the limitation of not being data producers, which makes them use sometimes opaque second-hand government data [Ton23], data journalists believe that sharing their choices, work methodology, and even uncertainty with the audience improve information clarity. According to the perception of 36 Brazilian data journalists, transparency is not only a way for them to communicate their work method but also a path to establish a relationship based on honesty with the readership and combat misinformation [Geh20].

A.2.2 *Data as a cue to perceived credibility in the (dis)information landscape*

Numbers, statistics, and data visualizations are potent cues in journalism and are often connected with straightforward communication of the facts. Fact-checkers also use data when verifying public claims, which implies that numbers are more accurate than discourse. Activating the same perception, mis- and disinformation narratives often use numbers and statistics to claim reliability.

Fabricated content related to the Covid-19 pandemic is an example of number manipulation as part of a misleading narrative. In a content analysis to explore 407 texts of false content that circulated during the first months of the pandemic in Brazil, Gehrke and Benetti identified that "data" was the third most recurrent category of the corpus analyzed, making up 19.66% of the cases. With the intent of minimizing the impact of the pandemic and arguing that the news media was creating terror, numbers and statistics were employed to construct a narrative that aimed to "demonstrate" that figures reported by the media were exaggerated. The examples included over-reported cases and deaths to the disease and allegedly empty hospitals.

Whereas pictures and videos are primarily adopted as evidence in disinformation narratives [DPD+21], numbers contained in data visualizations are part of what Peng, Lu, and Shen, p. 228 calls "visual features as arguments" when discussing visual features of misinformation posts that might influence people's credibility perception.

Given that visual mis- and disinformation has been studied less than general forms of manipulation, it is hard to estimate the frequency with which the layout of a news website is copied and converted into false content. Nevertheless, Peng, Lu, and Shen list that aesthetics usually work as a heuristic by providing people with hints that suggest (or not) the message come from a professional and credible source. Moreover, a previous study developed in our project found that news brands/logos are a powerful cue for people to assess the news' reliability [GEdVH24].

To exemplify this problem, we present fact-checked publications classified as "false" text and images that circulated online in 2022 mainly by mimicking the layout of news websites and logos of journalistic brands (see Figures 22 and 23). The content we use here was verified by fact-checking agencies that are signatories of the International Fact-checking Network (IFCN), which provides rigorous methodological and transparency premises that must be followed and shared with the audience.

Regarding Figure 22, the logo employed in the fabricated message (A) aims to attribute credibility to the content, by using the same shape and color as (B), which is the verified CNN Instagram account. Due to changes on Twitter (C), CNN's logo shape and verification batch are slightly different one year later. Still, a quick comparison between logos in different social media can easily generate confusion and mislead readers. The image and text (A) were verified and classified as false by the American fact-checking agency PolitiFact [Cur22].

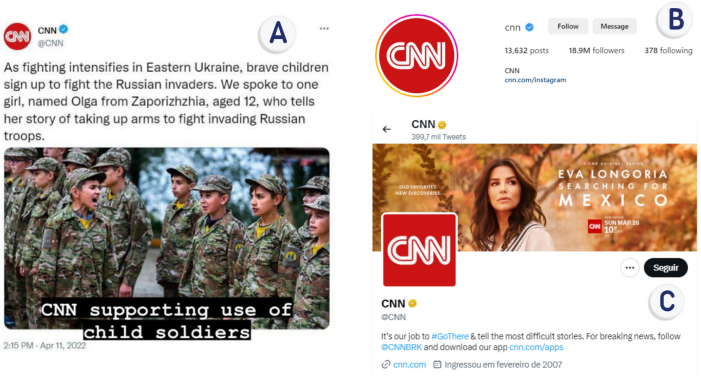


Figure 22: An Instagram post (A) with a screenshot of a manipulated tweet that was allegedly published by the American news organization CNN (B/C) on April 2022 concerning children's support of the Ukraine War.

Figure 23: Another type of manipulation consists of the complete copy (A) of a news website's original layout (B). In this example, the mobile version was adopted.

Figure 23 presents news falsely attributed to Deutsche Welle (DW) Brasil (A) in November 2022. Comparison it with the actual DW Brasil website (B), shows that a screenshot of the mobile website version was manipulated. The false text (A) claimed fraud in the Brazilian presidential elections and presented made-up statistics about the results. Besides the numbers, a fabricated news source with a "Ph.D. in Cybersecurity" was attributed within the text to contest the election results and mimic reporting procedures used in journalism, such as gathering information through sources. Brazilian fact-checking organization Agência Lupa classified the content as false [Sch22].

A.2.3 *Quotable signatures*

This section first introduces the technical aspects of quotable signatures, and then gives a practical description of the prototype, and some related considerations.

Digital signature schemes are a classical and widely used tool in modern cryptography [DH76; CMRR23]. In general, a digital signature scheme is a triple of algorithms KeyGen, Sign, and Verify. The algorithm KeyGen generates related pairs of a private and a public keys $(\mathsf{sk}, \mathsf{pk})$. The algorithm Sign signs any message m using a private key sk. This procedure produces a signature $s = \mathsf{Sign}_{\mathsf{sk}}(m)$ for m. The algorithm Verify verifies a message m and a signature s using the public key pk. Ignoring technicalities, the verification is successful only if *the signature was generated using m and the private key corresponding to* pk, *and neither the message m nor the signature s was altered.* We say that s is a signature for m signed with the private key sk. In other words, a secure signature scheme essentially ensures that only an entity in possession of the private key sk can produce a signature s for a message m, while the signature can be verified by anyone in possession of the public key pk.

This construction means that digital signatures ensure that (1) the message comes from a party that has a specific private key (identity), (2) the message has not been altered (integrity), and (3) a signer cannot lie about not signing a message, while also claiming that their private key remains private.

A newer concept is *quotable signature schemes* [KNSS19], which has been expanded upon by the authors [BELN23]. Summarizing, the main parts are as follows. A quotable signature scheme can be defined as digital signature schemes with an additional algorithm Quote. Given a message m and a quotable signature s, any third party can use Quote to extract a second quotable signature s' for a quote q from m, without knowing the secret key used to sign m or interacting with the party that signed m. This quotable signature s' is still signed with the private key used to sign m, and hence authenticates the original signing party as the author of the quote. In addition to having all the properties of standard digital signatures, quotable signatures also allow deriving where parts of the message have been removed relative to the quote. A signature for a quote is again a quotable signature with respect to sub-quotes of the quote.

In the Trust and News Authenticity project, we have developed a prototype of a tool which aims to mitigate the effect of disinformation by authenticating quotes from articles using quotable signatures. This authentication is intended to complement the already existing flagging of problematic content. Since quotable signatures do not verify the truthfulness of the content, but rather authenticate its origin and inte)grity, this approach is different from fact-checking. Essentially, rather than aiming to prove that the statement is correct, it validates that a statement is extracted *ipsis litteris* from its provenance without falsification. Figure 24 illustrates the user journeys when using the prototype.

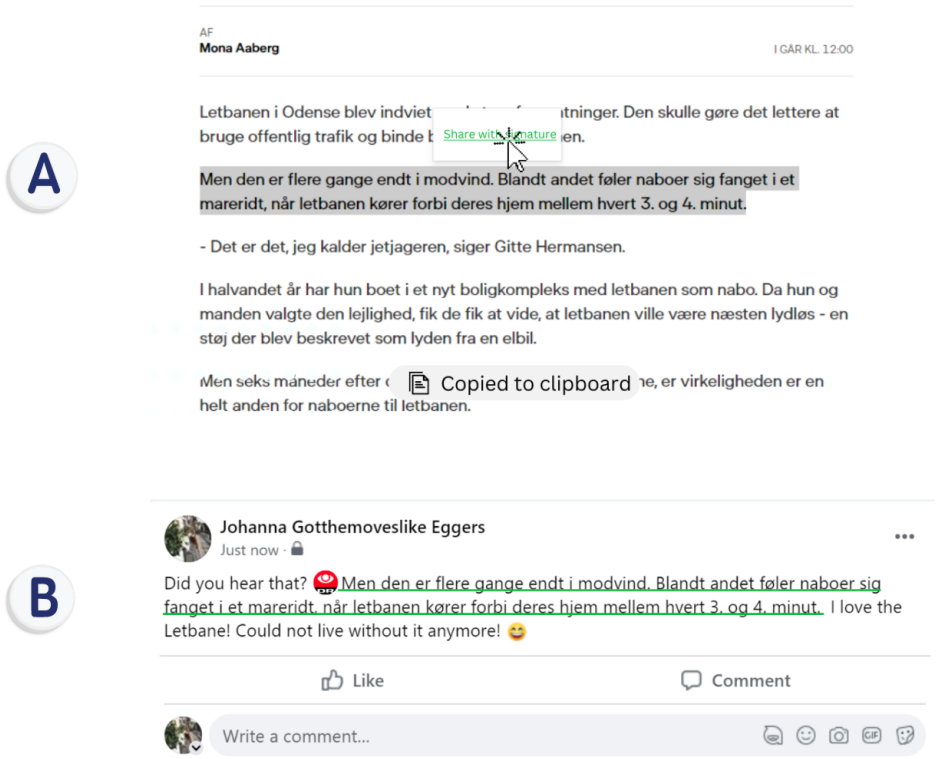


Figure 24: The user's journey when (A) reading and copying part of the news and (B) sharing the content on social media.

Figure 24 also shows that only the exact part copied and pasted is turned green (B), indicating that the quote, and only the quote, was authenticated, and that the excerpt comes from the original news media from which it was retrieved. The text added by the user is not highlighted. For our prototype, we used Facebook as social media since our project development occurred in Denmark, where 72% of the population use this platform for general purposes and 35% for news [NFR+22]. In addition, Facebook is browser-friendly, compared to other social media that prioritize app use.

The logo before the highlighted text (B) refers to DR, a public broadcaster highly trusted by the population [NFR+22], from which the quote originates. We decided to incorporate logos after our first project study with young Danes which indicated that news brands are, as a whole, a powerful cue for people to assess if a piece of news is trustworthy [GEdVH24].

By clicking on authenticated content, a reader can summon a popup with more information, such as who signed it, when it was signed, an indication of where text was removed, and a link to the original article. Additionally, information about what (quotable) signatures are and what being authenticated means can also be provided bythis popup.

Relating the prototype to the general terms of quotable signatures, the original source of the quote (for example an article) is the message, and the author or distributor of the article (a news outlet, for instance) is the

signing party. The party sharing the quote is the one extracting the signature for the quote, and the one reading the verified quote performs the verification. In practice, the act of extracting and including the signature for the quote, and of verifying the signature, would be completely automated, and happen in the background, requiring no additional user interaction.

Our prototype is separated into two parts: a library that can be used by media companies to sign their articles and a browser extension that allows users to quote with signatures and to verify signatures for quotes. The library contains implementations of the relevant algorithms, and it is intended that media companies can integrate it in their publishing workflow. The browser extension modifies websites such that both full articles and quotes with verified signatures are shown to be signed. It also allows the user to make quotes that include a signature when quoting from signed text. In addition, the browser extension provides more information for a quote to the user.

For our proposed approach to be effective, it would need to be widely adopted by news media, social media, and by users sharing and reading quotes from articles. Notably, if news media and social media integrate this solution into their websites, our approach can be employed without any explicit user awareness. With such integration, when a user copies a quote from a signed article, a signature for the quote is automatically generated, and an element including both quote and text is put into the clipboard, together with the plain text quote (in practice, this would be a `text/html` element and a `text/plain` element). When the user then pastes the quote, a website supporting signatures will use the clipboard element with a signature [W3C21].

A.3 DISCUSSION AND CONCLUSION

Adding quotable signature schemes to the data journalism transparency repertoire might help to reduce mis- and disinformation spread. In this exploratory and descriptive study, we argue that fabricating content and falsely attributing it to news websites is a current problem that can mislead readers and, in the end, undermine journalism the perceived credibility .

Even though we have mapped cases in which legacy news media are mainly the object of manipulation, we argue that the disinformation impact can quickly be extended to highly specialized news coverage websites, such as data journalism. Since this data-driven practice deals with massive amounts of data, analysis, and visualization, it can quickly become a target of mis- and disinformation narratives once it provides cues people usually trust, such as statistics. Thus, quotable signature schemes are particularly promising to data journalism initiatives. Furthermore, emerging forms such as predictive journalism [Dia22] could go in the same direction and suffer the consequences of a manipulative information environment.

Besides working as a resource to trace the origin of a text excerpt, quotable signatures schemes could be extended to validate pictures, videos, databases, and combinations of different formats. It would mean that data journalism, and other forms of journalism, could validate the provenance of different sources (including multimedia content) within news articles. Available to the readership, it improves and extends the transparency repertoires. Fact-checking agencies can also benefit from the same authentication structure by providing quotable signatures in pieces of verification to their readers. When analyzing claims, these agencies provide evidence of how they have checked them by providing the original sources used in the verification process. The method performed is crucial for fact-checking agencies to classify a claim as "false."

To make a difference in the future, media companies and users on social media need to adopt these quotable signatures. To have the best effect, social media platforms and news outlets should directly support quotable signatures, and the required extension should be natively integrated into browsers. Ultimately, this is also a way for platforms to engage in efforts to combat disinformation and protect democracy actively.

A.4 ACKNOWLEDGMENTS

We thank our colleague and project member Johanna Eggers for visualizing the user journey in Figure 24.

A.5 FUNDING

The research project Trust and News Authenticity develops a digital signature attached to journalistic content and a recognizable label for users to see if the content is authentic and verified. The project and, therefore, the present study received financial support from TrygFonden and the Digital Democracy Centre at the University of Southern Denmark.

B

IMAGE AUTHENTICITY IN THE AGE OF AI: DIGITAL SIGNATURES AS A DEFENSE AGAINST VISUAL DISINFORMATION

Johanna Eggers, Simon Erfurth, and Marília Gehrke. Image authenticity in the age of AI: digital signatures as a defense against visual disinformation, 2025. Submitted to Cambrdige Disinformation Summit 2025.

B.1 INTRODUCTION

With the rise of generative AI technology, it becomes increasingly difficult to accurately identify real and fake images in the online news environment, which leads to a threat to democracy and skepticism about the authenticity of (real) images [EH23]. By authenticity, we mean (online) content, which remains fundamentally unaltered and whose provenance can be traced. We emphasize that content being authentic should not be confused with content being "truthful", which is an orthogonal property. Accordingly, our approach is different from the one employed by fact-checkers, who assess a claim's veracity. In this conceptual and solution-driven paper, we offer the conceptual pillars for a novel approach to the well-known labeling systems so far used in mis- and disinformation mitigation. Our contribution, thus, lies on offering digital signatures to verify the authenticity of images. In doing so, we combine the knowledge and expertise gathered in media and journalism studies and cryptography.

Specifically in the context of images, the use of generative AI in the fabrication of multimedia content, often called deepfakes [PD19], is proving to be a particularly concerning type of visual disinformation [WL22]. In an electoral context, deepfakes include not only images, but also robocalls and synthetic videos that "steal" a politician's voice and/or image. Due to the concerns about the impact of fabricated content on the European elections in 2024, the European Union has called on technology platforms to outline plans to tackle deepfakes [OH24]. Similarly, fact-checking media organizations have planned joint actions to systematize claim verifications related to topics such as the European elections and the climate crisis [EFC24].

A common approach to mitigate the threats of AI generated fake images is to detect and label the fraudulent ones [KFL23]. This approach of negative labeling is still being developed and remains a computational and algorithmic challenge [MSLL21]. Additionally, warning labels potentially lead to backfire effects, e.g., the overall decrease of news credibility [vdMHO23]. With a focused literature review we map out the status quo of automated labeling and identify potentials before we introduce a

novel approach to image signing and positive labeling, e.g., marking authentic content. We argue that this approach benefits individuals and media publishers, including fact-checking organizations, and is less prone to technological errors and backfire effects.

Since images have received the status of evidence, those whose aim is to create and spread misleading information have tried to imitate news organizations' website layouts. We argued elsewhere that variations of digital signature schemes are particularly promising for reducing deception [GE23], including visual disinformation. Digital signature schemes are a classical and widely used tool in modern cryptography [DH76; CMRR23]. Ignoring minor technicalities, a digital signature scheme allows parties to sign documents with a secret key, which can then be verified by other parties possessing a corresponding public key. The verification is successful only if the signature was generated using the claimed message and the private key corresponds to the public key, and neither the message nor the signature was altered. We present a variation of digital signature schemes for images, which allow JPEG compression [Erf24]. Having this additional property allows the same digital signature to follow the image on platforms, such as social media, where it was not previously possible, due to it being unfeasible to store images uncompressed. This way, our digital signature scheme is a novel approach to follow images as they are distributed through the online information environment, while consistently present provenance and authenticity to users.

B.2 LITERATURE REVIEW

B.2.1 *Democratic lenses on news authenticity*

Our work is theoretically embedded in the normative perspective that a healthy democracy depends on well-informed citizens who will act based on factual information [PLS23]. Following professional values and ethical guidelines, journalism verifies and delivers accurate and authentic information to society through traditional practices and fact-checking. The latter has been defined as a new style of political news based on truth-seeking and holding public figures accountable [Gra16], which departs from the underlying assumption that disinformation – i.e., fabricated content that aims to cause deception – and lies can be sorted and differ from political disagreement [LEC+24].

In line with previous work, we define authenticity as "(...) a social construct and an act of performativity negotiated among actors" [GEdVH24, p. 4]. When applied to news, this concept refers to genuine content, that is, multimedia outcomes with their original purpose and set- up preserved in all stages from news production to distribution. Within this multimedia content, journalism has historically employed photography as evidence – which removes the journalist as a "(...) middleman between event and reader" [BC23b, p. 35], extrapolating from the use of photography as a representation of reality, but as reality itself.

The materiality of photography and video in journalism and social media platforms stimulates audiences to initially consider such objects genuine and, therefore, authentic. [BC23b, p. 37] have described authentication as a "(...) technique that is called to account whenever we encounter the machinic, the digital (...)," which comes into play when suspiciousness over content is raised. A recent study of how a group of young adults authenticated news in Denmark revealed that participants' high levels of trust in the news media made them use national journalistic brands to authenticate news they encountered on social media [GEdVH24]. That is, they would check if shared content was at all covered by legacy and public service media. Since journalism is not generally expected to deceive, actors who create and spread disinformation narratives with a deceptive intent often imitate authentic news websites' design[BM22]. For instance, credible news websites usually structure news by employing headlines, photos, loglines, and full text as part of their layout, which is commonly replicated in fabricated content[GE23].

Since part of visual manipulation is rooted in replicating the logo and the design of news websites and modifying its content, compromising the authenticity of online content, we argue in this paper that digital signatures are the most suitable approach to avoid deception.

B.2.2 *The power of images*

A picture is worth a thousand words. This universal saying derives from the observation that a picture or image can convey complex ideas, emotions, or messages without needing to put the same in written or verbal form. For example, images play a significant role in conveying news stories and information-seeking intentions [VHS20]. With the rise of AI image generators, can we still rely on images that rapidly spread online to convey trustworthy and authentic messages?

Images are not only a complementary part of a narrative and entail more than visual representations. They are an essential part of understanding and passing on knowledge and historical memory [Rüs06]. Images connect past and present and enable the cultural and historical continuity of a society [HS10]. The photograph had a special relationship to history, as it used to serve as proof for a specific event in time [HS10]. Nowadays, this relationship has changed. The role of photographs has been relativized quite quickly, as they can be staged and taken out of context. Photographs need to be authenticated and fact-checked carefully in order to keep their role of proving reality. Today, the rapid development of AI-generated images not only relativizes the previous role of photographs but completely transforms it. Photorealistic AI-generated images are taking advantage of the leap of faith that recipients still attribute to authentic photographs to add emotions to the (false) narrative. The text-to-image generators make the production of AI images universally accessible and resource friendly without users needing any special skill set.

The image itself is powerful in conveying messages because of the cognitive processing it evokes as well as the physiological response [BPBT06]. Images with e.g. threatening, intense or emotional elements lead to a stronger physiological response that indicates a higher engagement and relevancy of those images [BPBT06]. High-quality AI-generated news images have been shown to evoke similar emotions as human selected images [PBN+23] making them a potential substitute for traditional images and a strong influence on how the newsworthiness is evaluated by the recipients.

AI generators are not only used to produce misleading or harmful content, reputable news sources are using AI images in their editorial processes as well [Hau24; TTM24]. The AI image generators pose numerous opportunities and challenges for the news production, while ethical issues such as algorithmic bias are still broadly considered in research [TTM24]. An image in the traditional sense can entail a free interpretation of a message, such as a drawing, painting, or illustration (analog or digital). It can also be a realistic representation of the same, describing the concept or event accurately. Both of those versions can be created by AI and have potential for efficient production of high-quality and unique news stories. Recent research shows that news users perceive the usage of AI generated generic images or illustrations for a news article more positively but the usage of photorealistic AI images, if e.g. a real photograph did not exist, more negatively [FN24]. The bad reputation and high attention that maliciously created AI images receive seem to influence the evaluation of AI generated images overall. Still, as we learned from previous disruptive technological advancements: once invented, they stick around. Therefore, we can expect the perceived authenticity of AI generated images overall to develop over time.

When comparing the current state of AI image editing and creation in both disinformation entities and reputable sources, we can observe that news creators are currently still holding back with using images that were solely produced by AI, except when they are reporting about AI. Typical current news coverage including AI are comparisons, emphasizing the similarity of e.g. photographs and photorealistic AI images. The following images (Figure 25) were published in Politiken, titled "Artificial intelligence has made one of these images, but which one?(...)" [Kjæ24].

At first glance, the usage of pure AI images to report on other issues than AI seems low in news organizations. However, they are using established visual media companies and stock suppliers to find illustrations, pictures or videos to attach to their content. Those stock suppliers are increasingly promoting AI generated content. Getty Images, for example, launched a tool in collaboration with NVIDIA to provide text-to-image generators that create "commercially safe and legally protected images" [Ima24]. The creators on those platforms that create and edit pictures, illustrations and other media content are not subject to strict regulation or control over how their content is produced. They might

Figure 25: AI generated image, published by politiken.dk, 8th March 2024.

openly or covertly use AI generators for their content production. Ritzau is one of the platforms not yet promoting AI-generated content, while being aware that they don't have a reliable way to verify if their creators are using AI editors or generators. For the time being, they are relying on trust [Thy23].

The first examples show that also pure AI images can already make their way into news reporting. An AI image of a reconciliation between Prince William and Prince Harry surfaced prior to the coronation of King Charles III, see Figure 26. The image was created on Midjourney and originally published on medium.com while it was clarified that the image was produced by an AI generator and doesn't show real events. The creator announced that the purpose of this image was to "envision the possibility" of a reconciliation [tAI 24]. The image was virally used and requested by news outlets.

In conclusion, both reputable news sources and fraudulent entities are increasingly using AI editors and generators to produce images. Both real and fake images travel fast through online spaces. The issue with both being present in the information landscape is that it gets progressively more difficult to distinguish between them as the quality of the output equalizes with technological advancement. The solution is not only to identify that an image was AI-generated, but also if it was AI-generated with intent to harm.

Therefore, an important distinction is to be made, realizing that AI-generated does not categorically mean fake or bad. Even before the emergence of AI generators, journalistic images and photographs have not been neutral displays of reality. Press images and photographs can also become ambiguous by using specific angles, moments or framing narratives [Hau24]. The more important differentiation to make is who produced the image and news narrative, and highlighting the sources' authenticity. If a reputable news source uses either traditional or photore-

Figure 26: AI generated image created and published by medium.com, 29th April 2023.

alistic AI images, the same intention of informing the public of a certain concept or event can be assumed. The usage of the AI image does not take away from the journalistic quality.

B.2.3 *Visual mis- and disinformation*

Most scholarship related to misinformation and disinformation studies focuses on textual falsehoods despite the leading rates of visual-oriented social media [PLS23]. In one of the early examples of visual misinformation scholarship, Thomson et al. [TAD+20, p. 8] presented a typology of common operations in visual media, including manipulation techniques that involve modifying the picture itself—for example, changing the levels of saturation or artificial blurring—and changing the context of the picture at a source level, which they called "misattribution."

Research related to visual misinformation started gaining prominence during the COVID-19 pandemic (i.e., early 2020 onwards). In one of these studies, Brennen, Simon, and Nielsen [BSN20] analyzed the visual frames of narratives debunked by fact-checking organizations and found that, among 96 posts that contained images or videos, the main purpose of the manipulation (52%) was "serving as evidence" – that is, supporting untrue claims and narratives.

More recently, a group of scholars has pushed the debate toward visual misinformation due to changes in the prevalence of Artificial Intelligence (AI) and similar tools that make the fabrication of content cheaper, easy to believe in, and even easier to distribute. Whereas cheap fakes use less sophisticated technological advancement (e.g., photoshopping, lookalikes, speeding and slowing moving images), deepfakes include a more comprehensive range of audiovisual manipulation, such as face swapping and voice synthesis [PD19].

Many scholars are particularly concerned about deepfakes, broadly defined as synthetically created media. Weikmann, Greber & Nikolaou [WGN24], for instance, mapped two main reasons for concern: 1) audiences no longer believe that audio-visuals (broadly speaking) represent reality, and 2) people no longer trust they can discern between real and fake. Such outcomes might present challenges for journalism and democracy, given that "reading" cheap and deepfakes as authentic evidence can justify physical, sexual, and political violence [PD19]. This is particularly concerning for women and women of color, who are subject to political violence based on gender and race stereotypes.

In a study that examines the prevalence and characteristics of synthetic media on social media platform X (formerly known as Twitter) from December 2022 to September 2023, Corsi, Marino & Wong [CMW24] found an increase in AI-generated media, with a spike that followed the release of the fifth version of Midjourney – a generative artificial intelligence program used to create images first released in mid-2022. They identified 556 unique tweets containing fabricated images or videos viewed or watched 1.5 billion times. Quantitatively, most of the synthetic media found by the authors were non-political and non-malicious. Still, the existence of political figures as targets, they argue, raises concern about the potential misuse of this kind of technology.

Moreover, even the (apparently) non-political AI-generated images might be subject to political use. In the aftermath of the hurricanes that affected the population of the United States in 2024, "inoffensive" AI-produced images, such as a puppy and a child being rescued in a boat, were used as "proof" by conservative politicians who wanted to blame the Biden administration for not preventing disasters [Ahm24]. A similar phenomenon occurred in May 2024 after the floods that affected the South of Brazil. Fact-checking news organization Lupa verified a viral image of a man crying with a child in his arms (Figure 27). This AI-generated picture circulated on social media with the message, "May God take care of all the people of Rio Grande do Sul" [Fag24]. Even though there is no clear political claim related to this image, the green and yellow colors of the national flag – visible in the man's t-shirt in the picture – used to be associated with far-right demonstrations led by former president Jair Bolsonaro, including the Brazil Congress invasion on January 8th, 2021.

Nonetheless, it does not mean people are misled by deepfakes only or do not engage with cheap fakes. Because cheap fakes usually have actual footage as the starting point, they can be persuasive and effective, offering a "close representation of reality" [HvdMV24]. In a study that analyzed a sample of 2,500 images that circulated in political public groups on WhatsApp in India, Garimella & Eckles [GE20] detected that images taken out of context – thus, cheap fake – represented 34% of the misinformation image dataset followed by memes with fake quotes or statistics (30%).

Figure 27: AI-generated image of a man wearing the Brazilian national symbol on his t-shirt was verified by Lupa fact-checking organization.

One of the frequent forms of cheap fake manipulation is the use of lookalikes, usually authentic photos or footage that recirculate out of context. At the end of July 2024, the website Snopes labeled as "misleading" the image that allegedly portrayed United States vice-president Kamala Harris dancing on an episode of Soul Train (Figure 28), a TV show that aired from 1971 to 2006 [Lil24]. The video was, in fact, from a 2005 clip of Mariah Carey's song "It's Like That".

Though it is impossible to reveal what was the intention behind such a false narrative, the use of lookalikes is one of the well-known strategies adopted by those who want to attack women's reputations based on gender and race stereotypes, often suggesting that such behavior is not adequate for female politicians [Geh23].

B.2.4 *How fake images affect news authenticity*

Although not inherently malicious, AI generators pose new risks for the media ecosystem and perceived authenticity of news. The user-friendly interfaces and intuitive functionalities reduce the skills and resources needed to create (photorealistic) images. Therefore, many more sources with intent to harm can produce fraudulent content and spread it online. Typical techniques for disinformation spreading are simplified and accessible, such as impersonation of reputable sources. This can happen by creating fake websites that mimic the original [HP24], or creating fake social media accounts to resemble authentic accounts of e.g. politicians [ZFC19]. Another technique is the manipulation of authentic photographs [Ham24], making the depiction of the fake elements even more difficult. A prominent example of manipulated photographs is a picture

Figure 28: Footage was taken out of context to place US vice-president Kamala Harris falsely. This content was verified by Snopes.

from the immediate aftermath of the attempted assassination of Donald Trump on July 13th, 2024. The picture had been manipulated to show smiling Secret Service agents [Che24]. This image was then used to propagate conspiracy theories about the attack [Che24].

The visual aspect of images makes any news story more attention grabbing [Ham24]. False narratives that are used for propaganda purposes often rely on sensational and negative stories and images to catch even more attention from the recipients. The AI generators can now create fake images that defy the boundaries of what is physically possible or imaginable and are therefore an ideal accelerator for propaganda purposes [CC18]. This combination of propaganda and vivid, intense imagery creates a critical dynamic, as it can manipulate viewers' perceptions and emotions more effectively than before.

The exploitation of the different disinformation tactics used when creating fake images becomes especially prevalent on social media news feeds, where sensational and emotionally engaging content is more likely to be broadly shared [HKS24; PPSS23]. Additionally, more and more users enter social media via their mobile devices [NFR+24], where the screen is small, and the attention span is short. As follows from the peripheral route of the Elaboration Likelihood Model, low cognitive processing effort when scrolling through news most likely leads to news users paying little attention to details of the images and a low probability to self-initiate authenticity checks [PC86].

The spread of images enhancing disinformation, manipulation and propaganda can also impair the public trust in reputable news sources and institutions. When manipulated or fake images are used to display false narratives, it can lead to skepticism about the authenticity of all media content, making it difficult for audiences to identify what is real and what is fake [Ham24]. AI images that intend to harm therefore lead to an undermining of trust in authentically captured media.

B.2.5 *Label approaches and effects*

When acknowledging the importance of images for news narratives and authenticity, it is also crucial to determine when and how to protect image authenticity, with AI-generated images spreading rapidly in online environments. The AI images spreading online are not inherently wrong or bad, but the production and distribution patterns show broad concerns for disinformation and scam production.

The most common approach to directly combat false information that has already been shared is to identify and flag fraudulent content by e.g., the usage of warning labels [KFL23]. While this labeling approach is widely used, it remains a computational and algorithmic challenge. Labeling all of the false content online is naturally unfeasible and the threat of censorship occurs with automation approaches [MSLL21]. Warning labels have shown to potentially lead to negative backfire effects, such as a general distrust in (legitimate) news and deception bias, meaning that news readers are primed to assume that the information they encounter is more likely to be false than true [vdMHO23]. The protection of images attached to online news therefore leads to technological and conceptual challenges for which a new standard still needs to be created. The protection of images that represent or are attached to news content therefore poses not only technological but also conceptual challenges that have not yet been sufficiently met by negative labeling through fact- checking instances. It is time to develop a new standard that can adequately protect the integrity and authenticity of images used in digital journalism.

The challenges of automating and strategically counteracting AI generated images lie in the fact that AI generated images that spread in online environments are highly scalable [NNY+24]. Disinformation scales up easily due to effortless access to advanced technologies such as AI generators and beneficial distribution possibilities on social media, where it reaches a wide audience quickly [NNY+24].

Recent developments in counteracting AI-generated disinformation in general are most commonly focused on detecting synthetic media by using a variety of methods [BPT+24]. The recent systems developed move away from negative labeling (warning labels) to e.g. specifically detect and label AI-generated images on social media [PNC24]. By using approaches such as frequency analysis technologies and machine learning algorithms, these systems can detect AI images and differentiate them from non-AI images [PNC24]. While performing an important task and showing a high level of innovative response to AI images, the problem of distinguishing AI images with the intention to harm from AI images used in journalistic manners remains unresolved.

By introducing digital signature schemes that authenticate the real content when it is shared online, we address this problem by making the real and fake images distinguishable again. This approach of following authentic images online and labeling them as such, relies on a technologically effective, automatable and innovative approach but also on positive

labels leading to the desired effects. Studies on established consumer labels have shown that negative labels generally have a stronger immediate effect on the recipient than positive labels [GDB04]. A positive label is expected to have a smaller direct effect on individual news consumers and their behavior online, but when applied on a broad scale, its impact can extend to nearly everyone. Even subtle changes, such as scrolling past fake content more quickly and spending slightly more time engaging with real content, can have a significant effect on cognitive processing and news consumption online.

B.2.6 *A Digital Signature Allowing JPEG Compression*

B.2.6.1 *Digital Signatures*

One promising way to add authenticity markers to online content, is to use the classical cryptographic primitive digital signature schemes [DH76; CMRR23]. Using digital signatures, content can be equipped with markers that mathematically guarantee the authenticity of the content. Before we describe the variation of digital signatures we propose using, we introduce digital signatures.

In a nutshell, digital signatures allow parties to *sign* messages in a way that other parties can verify the *signature* for the message, and be confident that the message originates from the claimed party and has not been modified. In practice, this works by generating a key-pair, consisting of a private key and a related public key. The private key allows *signing* messages, which generates a signature for the message being signed. The public key can be distributed freely,[1] and any party can verify a message against a signature and a public key, checking that the signature has been generated for this specific message, using the private key corresponding to this public key. This procedure guarantees three properties. (1) The identity of the signer is guaranteed, in the sense that it is impossible to forge a signature as being signed with a secret key, without actually possessing that secret key. (2) The integrity of the message is guaranteed, meaning that the message being verified is bit-for-bit identical to the message that the signer signed. And finally, (3) non-repudiation; if a signer has signed a message, the signature binds the signer to the message, in the sense that it is impossible for the signer to claim that they did not create this signature for this specific message (while still claiming their secret key remains secret).

The three outlined properties, make digital signatures a very strong tool for attaching authenticity labels to online content. Essentially, quality content creators can sign their content, and the digital signature can then follow the content as it is shared online, serving as an authenticity label, which guarantees the origin and integrity of the content. How-

1 Infrastructure for distributing public keys in a way one can be assured of the identity of the owner of the public key exists, and is widely used. It is referred to as public key infrastructure [Uni19].

ever, one issue with using standard digital signatures for this is the strictness of Property (2). This property requires the message being authenticated to be bit-for-bit identical to the message that was signed, which, in essence, means that articles and images can not be modified in any way. For example, articles can not be quoted, and images can not be compressed. While this could be somewhat alleviated by having the signer sign multiple predetermined versions of the content (for example select quotes or important paragraphs from text, and compressed or cropped versions of images), such a solution would either be limited in the number of transformations it support, or incur exponential costs.

To resolve the issue with minimal overhead costs, we developed (somewhat) homomorphic digital signatures, which allow a specific type of transformation to be performed freely on the message, without invalidating the signature (both for text and images). For text, a natural transformation to support is quoting parts of it. We developed digital signatures that are homomorphic with respect to quoting in [BELN23; GE23].

For images, there is a wider selection of natural transformations that could be supported, such as cropping, rotating, filtering, compressing, and many more. We observe, that when an image is uploaded to a social media, it is compressed, and hence, compression might be the most commonly performed image transformation, and we focused on developing a digital signature scheme for images, which is homomorphic with respect to (JPEG) compression [Erf24].

B.2.6.2 *JPEG Compression*

To describe how the signature scheme allowing JPEG compression is constructed, we first give a brief introduction to how JPEG compression works. More details can be found in [Wal91; Erf24]. JPEG compression uses how the human vision system works, and preserves more of the details that humans are good at noticing, while discarding more of the details humans are bad at noticing. Concretely, humans are more sensitive to changes in color than changes in brightness, and humans are more sensitive to low frequency changes in intensity of colors/brightness than to high frequency changes in intensity of color/brightness. Hence, JPEG compression preserves more information about brightness and low frequency changes than it does about color and high frequency changes. In practice, the compression is done with the following steps:

1. The image is converted from RGB color space to YCbCr color space, meaning that rather than having three channels representing how much red (R), green (G), and blue (B), respectively, there is in each pixel of the image, it has three channels where one (Y) represent how bright each pixel is, and the other two channels (blue-difference (Cb) and red-difference (Cr)) represents the color of each pixel. In later steps, this allows preserving more information for the brightness channel than for the color channels.

2. An optional down-sampling step, which we do not apply.

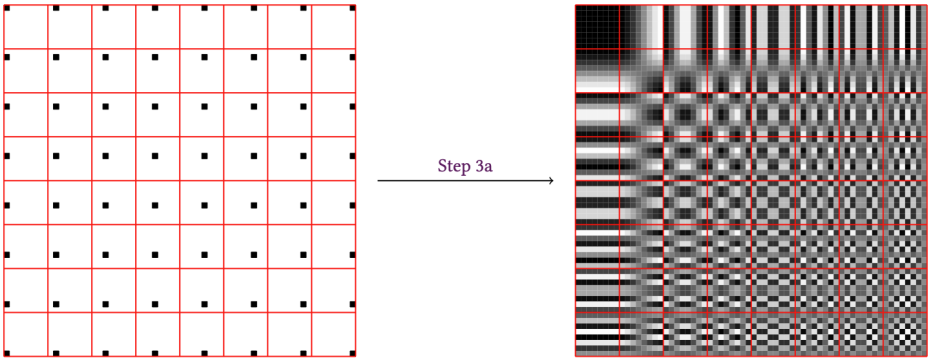


Figure 29: Illustration of how each coefficient goes from representing one pixel in an 8 × 8 block, to instead representing a DC wave over the block. Figure originally appeared in [Erf24].

3. The image is split into 8 × 8 pixel blocks, which are processed individually for each channel in the next 2 steps.
   a) The discrete cosine transformation (DCT) is applied to the block, such that each pixel in the block now represents how much of a discrete cosine wave there is in the entire block, rather than just representing the intensity of the specific pixel. This is illustrated in Figure 29, where it can be seen that each pixel in the 8 × 8 block goes from representing just itself (seen on the left) to representing a discrete cosine wave over the entire block (the "wave" of each pixel shown on the right). This step allows the next step to preserve more information about the low frequency waves (shown in the upper left corner of the right side of Figure 29), and less information about the high frequency waves (shown in the lower right corner of the right side of Figure 29).
   b) The block is quantizised, meaning that entry in the block is divided by a value from a *quantization table*, and then rounded. A quantization table is an 8×8 table, with values, that specifies how much information each entry in the block can "keep". The larger the value in the quantization table, the less information is left after rounding. The parameter for a JPEG compression contains two quantization tables, one for the brightness channel and one for the color channels. This is the step where more information is preserved for details the human vision system is good at noticing. First, the quantization tables contain larger values in entries corresponding to high frequency discrete cosine waves, than in entries corresponding to values representing low frequency discrete cosine waves. Additionally, the values in the quantization table used for the color channels are generally larger than the values in the quantization table used for the brightness channel.

4. The entire image is encoded using a (lossless) entropy encoder. This step does not affect our construction, so we do not provide details.

B.2.6.3 *Signature Construction*

Returning to the problem of digital signatures requiring a bit-for-bit identical image, it can be observed that the only step where information is lost is Step (3b), when rounding is done. Thus, our goal for [Erf24] was to create a signature scheme that allows providing some extra information, which, during verification, can "replace" the information lost during compression, and allow the signature to still be verified. We made the key observation that when the value in the quantization table is a power of two (1, 2, 4, 8, 16 and so on), dividing and rounding is equivalent to truncating some of the least significant digits. That is, the least significant digits are removed, but all other digits stay the same.

This observation led to the following idea: to sign an image, first create a digest related to the entire image, but with the additional property that it is possible to replace any number of least significant digits with some other information, and still obtain the same digest. Then, sign the digest, in place of the image.[2] The value that can replace the least significant digits when computing the digest, can be derived from the digits being truncated, and, in practical cases, takes up much less space than what is saved by compressing the image.

Using this idea, compression is done using quantization tables that contain only powers of two, and the value that can replace the digits truncated by the compression is added to the signature for the image, allowing verification to be done.

In order to create the digest, we use what is called a hash function, which is a special type of function with the following properties. (1) It is easy to evaluate on any input. (2) Given a random output value, it is infeasible to find input mapping to that output. (3) It is also infeasible to find two inputs mapping to the same output [Dwo15]. The entire process of creating a digest with the properties we desire is illustrated in Figure 30. Figure 30 shows how a digest represents the first entry in every 8×8 block in either the brightness channel or the color channels is created. Essentially, the least significant bit of all entries that are first entries in an 8×8 block are hashed together, then the resulting digest is hashed together with the second least significant bit of all the entries, and so on, creating a chain of hashes. The last link in the chain (resulting from hashing another chain link and all the most significant bits) is called the chain end. For each entry in the 8×8 blocks and for both color and brightness, a chain like the one in Figure 30 is created. As a final step, the ends of all the chains of hashes are hashed together, and signed using a standard digital signature scheme.

2 In practice, all digital signatures work by creating a digest of the message, which is then signed.

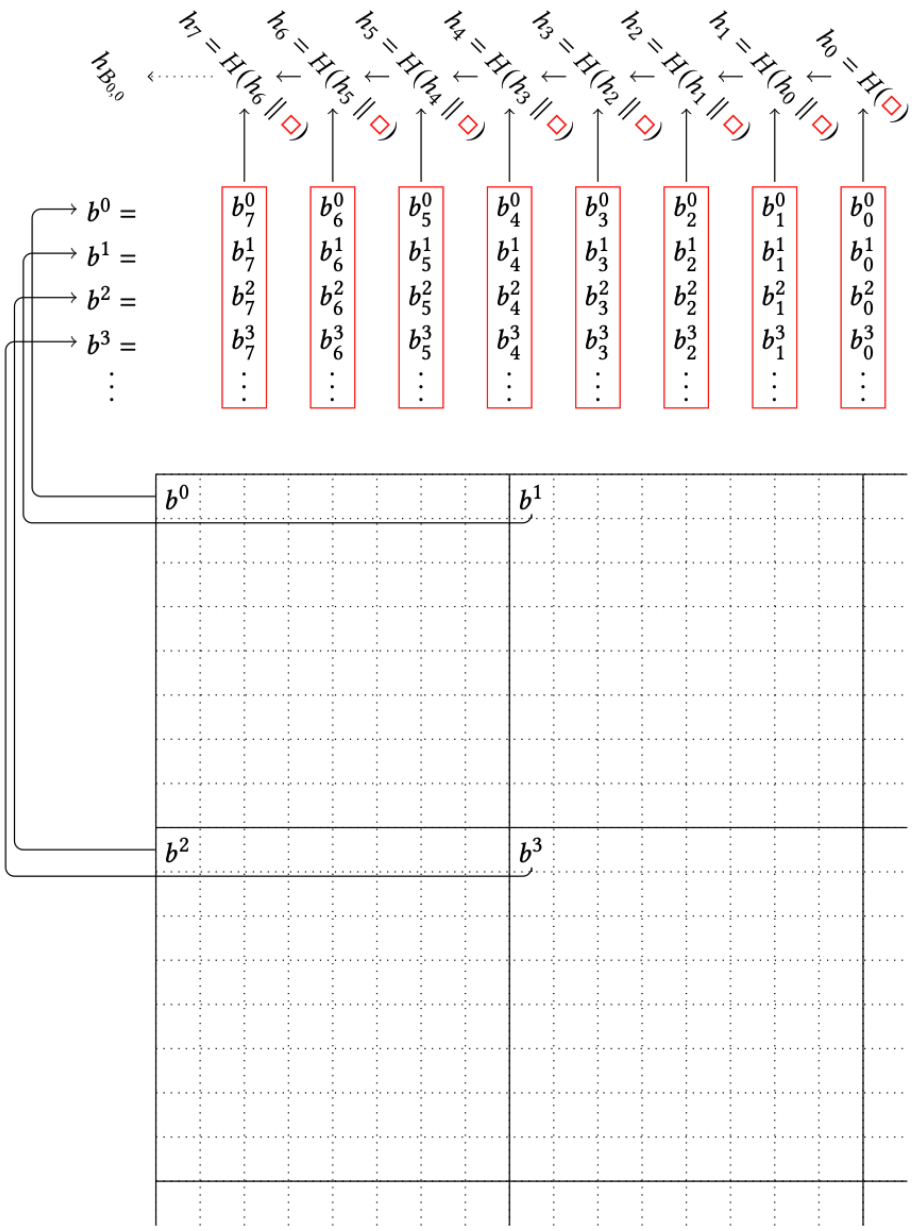


Figure 30: Creating the digest corresponding to one entry in the 8×8 blocks. Figure originally appeared in [Erf24].

When the image is compressed – and all the entries corresponding to the same entry in the quantization table are truncated by the same amount – just one link from each of the chains of hashes needs to be provided. When compressing, these links are calculated, and added to the signature. With an updated signature, one can now obtain the digest that was signed by recalculating the remaining links of the chains of hashes, and hashing all the ends together. If the image was compressed using quantization tables containing only powers of two, and the correct chain links were added to the signature, this digest is the value that was signed.

We claim that the signature scheme constructed like this is secure, in the sense that a signature for an image is never going to be a valid signature for an image that could not be obtained by compressing the original image. A formal definition of this notion of unforgeability and a proof of the security of the scheme can be found in [Erf24].

B.2.6.4 *Performance*

There are two performance aspects to consider with our signature scheme. One is if the signature scheme is efficient in terms of how much computation/storage it requires, and another is if compression done with powers of two is a good way to do compression, considering how much it reduces the size of an image versus how much it degrades the visual fidelity of the image.

To answer the first part, it can be observed that the signature before compressing has the same size as a standard digital signature, and after compressing, it additionally contains $2 \cdot 8 \cdot 8 = 128$ chain links, each of which is one hash digest. In [Erf24], we analyse how this lines up with different approaches, but the essential takeaway is that the overhead is minimal. For realistic parameters for both image size, hash function and standard digital signature scheme, an image compressed down to less than 5% of its original size has an overhead of just over 4%. For computation, the overhead is just $128 \cdot 8 + 1$ hash function evaluations, which is fast.

In order to compare the visual fidelity, we first tried simple ocular comparison, but with this approach, it was not possible to reliably tell apart images compressed using our approach and images compressed using standard quantization tables. Instead, we used two image metrics, both supposed to reflect how the human vision system perceives similarity[WSB03; ZZMZ11a]. We detail how the comparison was done in [Erf24], but in essence, for each image in a database of test images [PLZ+09], we found a quantization table with only powers of two, giving a compressed image with a size similar to the image compressed using a standard quantization table. For multiple different standard quantization tables, our approach resulted in an average size difference of less than 1 kB, and an average image quality metrics score difference of less than 0.1%, see Table 6 and [Erf24, Section 5].

Table 6: Comparison of compression done with regular compression parameters and compression done with parameters satisfying our approach. QF25/QF50/QF80 refers to the quality factor used for standard compression parameters.

		Size	MS-SSIM	FSIMc
QF25	**Our approach**	16.0 kB	0.960	0.978
	Unmodified	15.1 kB	0.959	0.978
QF50	**Our approach**	25.4 kB	0.979	0.991
	Unmodified	24.4 kB	0.979	0.991
QF80	**Our approach**	43.9 kB	0.990	0.997
	Unmodified	43.4 kB	0.991	0.997

B.3 DISCUSSION

In this paper, we presented the conceptual structure of a novel approach to verifying image authenticity through digital signatures. Our approach is complementary to fact-checking and other forms of mis- and disinformation mitigation. We argued that current most established measures to counteract on disinformation are not sufficient to address the growing challenges of AI image generators. The rise of AI content, especially deepfakes and photorealistic images, makes it increasingly difficult for recipients to distinguish between the good and the bad, as the quality equalizes. We introduced digital signatures schemes as a robust solution that is suitable by tracing back the origin of images shared online. Apart from the suitability we also highlighted the technological viability of this approach by guiding through the steps necessary to allow JPEG compression for images while still keeping the digital signature in place, which ultimately allows authenticated image sharing on social media platforms and similar intermediaries. The visual intervention of the authentication process for the images is a digital label attached to the image.

Figure 31 shows a mock example of how the label could be displayed to news consumers. We see a persona sharing a picture of the attempted assassination of Donald Trump into a social media news feed. The image is labeled as "source confirmed" (see the bottom right on the picture), emphasizing that the origin of the image is authentic and reputable. In this scenario, the image could have been shared directly from a website or post or downloaded and uploaded again within an original post.

In the next step, we aim to engage in the discussion of who gets to implement the digital signatures. One potential strand would be networks that congregate news organizations and/or journalists under the same ethical guidelines and codes of principle, meaning there is a process of best practices in the community and an attempt to hold each other accountable (e.g., Reporters Without Borders, the Global Investigative Journalism Network, and the International Fact-Checking Network).

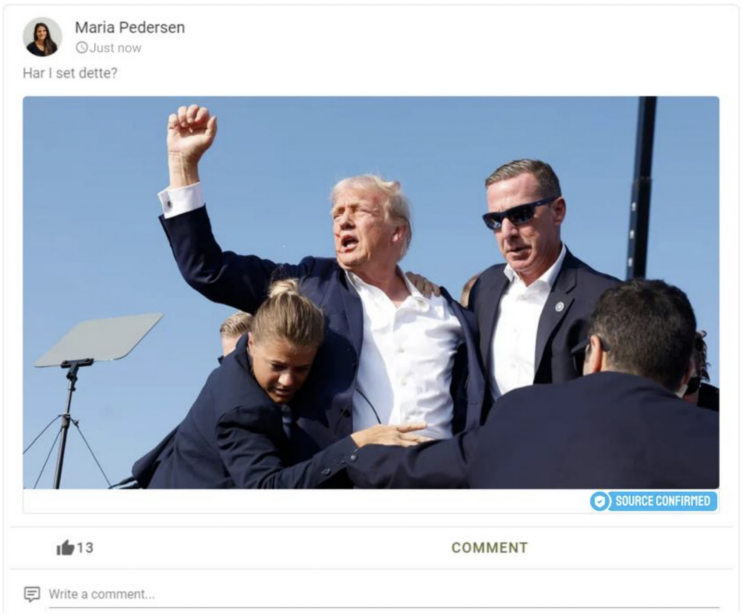


Figure 31: Mock Social Media post including the mock content label, produced by the authors.

C

PROPORTIONATE CONTRIBUTION TO PUBLICATIONS

According to the PhD school's guidelines for coauthor statements, the senior author on each publication (highlighted in bold) has evaluated my contribution as follows.

- **Joan Boyar**, Simon Erfurth, Kim S. Larsen, and Ruben Niederhagen. *Quotable signatures for authenticating shared quotes.*
  Intellectual input: 60%; Experiments/computations and/or data analysis: 100%; Writing process: 80%.

- **Joan Boyar** and Simon Erfurth. *Folding schemes with privacy preserving selective verification.*
  Intellectual input: 80%; Experiments/computations and/or data analysis: Not applicable; Writing process: 80%.

- **Marília Gehrke** and Simon Erfurth. *Adding quotable signatures to the transparency repertoire in data journalism.*
  Intellectual input: 40%; Experiments/computations and/or data analysis: 60%; Writing process: 40%.

- Johanna Eggers, Simon Erfurth, and **Marília Gehrke**. *Image authenticity in the age of AI: digital signatures as a defense against visual disinformation.*
  Intellectual input: 40%; Experiments/computations and/or data analysis: 40%; Writing process: 40%.

LIST OF FIGURES

[AAE+21] Fatima K. Abu Salem, Roaa Al Feel, Shady Elbassuoni, Hiyam Ghannam, Mohamad Jaber, and May Farah. Meta-learning for fake news detection surrounding the syrian war. *Patterns*, 2(11), 2021. DOI: 10.1016/j.patter.2021.100369.

[ABB+22] Jean-Philippe Aumasson, Daniel J. Bernstein, Ward Beullens, Christoph Dobraunig, Maria Eichlseder, Scott Fluhrer, Stefan-Lukas Gazdag, Andreas Hülsing, Panos Kampanakis, Stefan Kölbl, Tanja Lange, Martin M. Lauridsen, Florian Mendel, Ruben Niederhagen, Christian Rechberger, Joost Rijneveld, Peter Schwabe, and Bas Westerbaa. SPHINCS+. submission to the NIST post-quantum project, v.3.1, 2022. URL: https://sphincs.org/data/sphincs+-r3.1-specification.pdf.

[ABC+15] Jae Hyun Ahn, Dan Boneh, Jan Camenisch, Susan Hohenberger, Abhi Shelat, and Brent Waters. Computing on authenticated data. *Journal of Cryptology*, 28(2):351–395, 2015. DOI: 10.1007/S00145-014-9182-0.

[ACdMT05] Giuseppe Ateniese, Daniel H. Chou, Breno de Medeiros, and Gene Tsudik. Sanitizable signatures. In *Computer Security - ESORICS 2005*, volume 3679 of *Lecture Notes in Computer Science*, pages 159–177. Springer, 2005. DOI: 10.1007/11555827_10.

[ACF02] Masayuki Abe, Ronald Cramer, and Serge Fehr. Non-interactive distributed-verifier proofs and proving relations among commitments. In *Advances in Cryptology - ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 206–223. Springer, 2002. DOI: 10.1007/3-540-36178-2_13.

[Ado] Adobe. JPEG2000 files. https://www.adobe.com/creativecloud/file-types/image/raster/jpeg-2000-file.html. (Visited on 12/06/2023).

[AG13] Nicola Asuni and Andrea Giachetti. TESTIMAGES: A large data archive for display and algorithm testing. *Journal of Graphics Tools*, 17(4):113–125, 2013. DOI: 10.1080/2165347X.2015.1024298.

[AGB+22] Kevin Aslett, Andrew M. Guess, Richard Bonneau, Jonathan Nagler, and Joshua A. Tucker. News credibility labels have limited average effects on news diet quality and fail to reduce misperceptions. *Science Advances*, 8(18), 2022. DOI: 10.1126/sciadv.abl3844.

[Ahm24] Amal Ahmed. The hurricane conspiracies made it clear—we're going climate delulu, 2024. URL: https://atmos.earth/the-hurricane-conspiracies-made-it-clear-were-going-climate-delulu/ (visited on 12/12/2024).

[AJAZ22] Edward L. Amoruso, Stephen P. Johnson, Raghu Nandan Avula, and Cliff C. Zou. A web infrastructure for certifying multimedia news content for fake news defense. In *IEEE Symposium on Computers and Communications - ISCC 2022*, pages 1–7. IEEE, 2022. DOI: 10.1109/ISCC55528.2022.9912787.

[AO09] Adam L. Alter and Daniel M. Oppenheimer. Uniting the tribes of fluency to form a metacognitive nation. *Personality and Social Psychology Review*, 13(3):219–235, 2009. DOI: 10.1177/1088868309341564.

[BBB+18] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Gregory Maxwell. Bulletproofs: short proofs for confidential transactions and more. In *IEEE Symposium on Security and Privacy - S&P 2018*, pages 315–334. IEEE Computer Society, 2018. DOI: 10.1109/SP.2018.00020.

[BBD+10] Christina Brzuska, Heike Busch, Özgür Dagdelen, Marc Fischlin, Martin Franz, Stefan Katzenbeisser, Mark Manulis, Cristina Onete, Andreas Peter, Bertram Poettering, and Dominique Schröder. Redactable signatures for tree-structured data: definitions and constructions. In *Applied Cryptography and Network Security - ACNS 2010*, volume 6123 of *Lecture Notes in Computer Science*, pages 87–104, 2010. DOI: 10.1007/978-3-642-13708-2_6.

[BC23a] Benedikt Bünz and Binyi Chen. Protostar: generic efficient accumulation/folding for special-sound protocols. In *Advances in Cryptology - ASIACRYPT 2023*, volume 14439 of *Lecture Notes in Computer Science*, pages 77–110. Springer, 2023. DOI: 10.1007/978-981-99-8724-5_3.

[BC23b] Anthony Glyn Burton and Wendy Hui Kyong Chun. *Algorithmic Authenticity*. Meson Press, 2023. DOI: 10.14619/2102.

[BC24] Dan Boneh and Binyi Chen. Latticefold: a lattice-based folding scheme and its applications to succinct proof systems. *IACR Cryptol. ePrint Arch.*:257, 2024. URL: https://eprint.iacr.org/2024/257.

[BCC+16] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In *Advances in Cryptology - EUROCRYPT 2016*, vol-

ume 9666 of *Lecture Notes in Computer Science*, pages 327–357. Springer, 2016. DOI: 10.1007/978-3-662-49896-5_12.

[BCCT13] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive composition and bootstrapping for SNARKS and proof-carrying data. In *Symposium on Theory of Computing Conference - STOC'13*, pages 111–120. ACM, 2013. DOI: 10.1145/2488608.2488623.

[BCL+21] Benedikt Bünz, Alessandro Chiesa, William Lin, Pratyush Mishra, and Nicholas Spooner. Proof-carrying data without succinct arguments. In *Advances in Cryptology - CRYPTO 2021*, volume 12825 of *Lecture Notes in Computer Science*, pages 681–710. Springer, 2021. DOI: 10.1007/978-3-030-84242-0_24.

[BCMS20] Benedikt Bünz, Alessandro Chiesa, Pratyush Mishra, and Nicholas Spooner. Recursive proof composition from accumulation schemes. In *Theory of Cryptography - TCC 2020*, volume 12551 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2020. DOI: 10.1007/978-3-030-64378-2_1.

[BCR+19] Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. Aurora: transparent succinct arguments for R1CS. In *Advances in Cryptology - EUROCRYPT 2019*, volume 11476 of *Lecture Notes in Computer Science*, pages 103–128. Springer, 2019. DOI: 10.1007/978-3-030-17653-2_4.

[BCTV17] Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. Scalable zero knowledge via cycles of elliptic curves. *Algorithmica*, 79(4):1102–1160, 2017. DOI: 10.1007/S00453-016-0221-0.

[BDFG21] Dan Boneh, Justin Drake, Ben Fisch, and Ariel Gabizon. Halo infinite: proof-carrying data from additive polynomial commitments. In *Advances in Cryptology - CRYPTO 2021*, volume 12825 of *Lecture Notes in Computer Science*, pages 649–680. Springer, 2021. DOI: 10.1007/978-3-030-84242-0_23.

[BE24] Joan Boyar and Simon Erfurth. Folding schemes with privacy preserving selective verification. *IACR Communications in Cryptology*, 1(4), 2024. URL: https://eprint.iacr.org/2024/1530.

[BELN23] Joan Boyar, Simon Erfurth, Kim S. Larsen, and Ruben Niederhagen. Quotable signatures for authenticating shared quotes. In *Progress in Cryptology - LATINCRYPT 2023*, volume 14168 of *Lecture Notes in Computer Science*,

pages 273–292. Springer, 2023. DOI: 10.1007/978-3-031-44469-2_14.

[BFF+09] Christina Brzuska, Marc Fischlin, Tobias Freudenreich, Anja Lehmann, Marcus Page, Jakob Schelbert, Dominique Schröder, and Florian Volk. Security of sanitizable signatures revisited. In *Public Key Cryptography - PKC 2009*, volume 5443 of *Lecture Notes in Computer Science*, pages 317–336. Springer, 2009. DOI: 10.1007/978-3-642-00468-1_18.

[BFS20] Benedikt Bünz, Ben Fisch, and Alan Szepieniec. Transparent SNARKs from DARK compilers. In *Advances in Cryptology - EUROCRYPT 2020*, volume 12105 of *Lecture Notes in Computer Science*, pages 677–706. Springer, 2020. DOI: 10.1007/978-3-030-45721-1_24.

[BGH19] Sean Bowe, Jack Grigg, and Daira Hopwood. Recursive proof composition without a trusted setup. *IACR Cryptol. ePrint Arch.*:1021, 2019. URL: https://eprint.iacr.org/2019/1021.

[BGR98a] Mihir Bellare, Juan A. Garay, and Tal Rabin. Batch verification with applications to cryptography and checking. In *LATIN '98*, volume 1380 of *Lecture Notes in Computer Science*, pages 170–191. Springer, 1998. DOI: 10.1007/BFB0054320.

[BGR98b] Mihir Bellare, Juan A. Garay, and Tal Rabin. Fast batch verification for modular exponentiation and digital signatures. In *Advances in Cryptology - EUROCRYPT '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 236–250. Springer, 1998. DOI: 10.1007/BFB0054130.

[BKP20] Ward Beullens, Shuichi Katsumata, and Federico Pintore. Calamari and Falafl: logarithmic (linkable) ring signatures from isogenies and lattices. In *Advances in Cryptology - ASIACRYPT 2020*, volume 12492 of *Lecture Notes in Computer Science*, pages 464–492. Springer, 2020. DOI: 10.1007/978-3-030-64834-3_16.

[BL17] Xavier Bultel and Pascal Lafourcade. Unlinkable and strongly accountable sanitizable signatures from verifiable ring signatures. In *Cryptology and Network Security, CANS 2017*, volume 11261 of *Lecture Notes in Computer Science*, pages 203–226. Springer, 2017. DOI: 10.1007/978-3-030-02641-7_10.

[BM22] Thomas J Billard and Rachel E. Moran. Designing trust: design style, political ideology, and trust in "fake" news websites. *Digital Journalism*, 11(3):519–546, 2022. DOI: 10.1080/21670811.2022.2087098.

[BN02] Mihir Bellare and Gregory Neven. Transitive signatures based on factoring and RSA. In *Advances in Cryptology - ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 397–414. Springer, 2002. DOI: 10.1007/3-540-36178-2_25.

[Bor22] Ali Borji. Generated faces in the wild: quantitative comparison of stable diffusion, midjourney and dall-e 2, 2022. arXiv: 2210.00586 [cs.CV]. URL: https://arxiv.org/abs/2210.00586.

[BPBT06] Edward Bernat, Christopher J. Patrick, Stephen D. Benning, and Auke Tellegen. Effects of picture content and intensity on affective physiological response. *Psychophysiology*, 43(1):93–103, 2006. DOI: 10.1111/j.1469-8986.2006.00380.x.

[BPS17] Arne Bilzhause, Henrich C. Pöhls, and Kai Samelin. Position paper: the past, present, and future of sanitizable and redactable signatures. In *Availability, Reliability and Security - ARES 2017*. ACM, 2017. URL: https://doi.org/10.1145/3098954.3104058.

[BPT+24] Kalina Bontcheva, Symeon Papadopoulous, Filareti Tsalakanidou, Riccardo Gallotti, Lidia Dutkiewicz, Noémie Krack, Denis Teyssou, Francesco Severio Nucci, Jochen Spangenberg, Ivan Srba, Patrick Aichroth, Luca Cuccovillo, and Luisa Verdoliva. Generative ai and disinformation: recent advances, challenges, and opportunities, 2024. URL: https://lirias.kuleuven.be/retrieve/758830.

[BSN20] J. Scott Brennen, Felix M. Simon, and Rasmus Kleis Nielsen. Beyond (mis)representation: visuals in covid-19 misinformation. *The International Journal of Press/Politics*, 26(1):277–299, 2020. DOI: 10.1177/1940161220964780.

[C2P] C2PA. Coalition for Content Provenance and Authenticity (C2PA). https://c2pa.org/. (Visited on 12/12/2024).

[CC18] Anne K. Cybenko and George Cybenko. Ai and fake news. *IEEE Intelligent Systems*, 33(5):1–5, 2018. DOI: 10.1109/mis.2018.2877280.

[CCDW20] Weikeng Chen, Alessandro Chiesa, Emma Dauterman, and Nicholas P. Ward. Reducing participation costs via incremental verification for ledger systems. *IACR Cryptol. ePrint Arch.*:1522, 2020. URL: https://eprint.iacr.org/2020/1522.

[CF13] Dario Catalano and Dario Fiore. Vector commitments and their applications. In *Public-Key Cryptography - PKC 2013*, volume 7778 of *Lecture Notes in Computer Science*, pages 55–72. Springer, 2013. DOI: 10.1007/978-3-642-36362-7_5.

[Che24] Reuters Fact Check. Fact check: images of trump, secret service smiling after shooting are fake, 2024. URL: https://www.reuters.com/fact-check/images-trump-secret-service-smiling-after-shooting-are-fake-2024-07-14/ (visited on 12/12/2024).

[CKA] Allan Cheboi, Peter Kimani, and Justin Arenstein. How hate speech trolls targeted kenya's 2022 elections. URL: https://disinfo.africa/early-detection-and-countering-hate-speech-during-the-2022-kenyan-elections-e0f183b7bdd1 (visited on 03/06/2023).

[CMRR23] Lily Chen, Dustin Moody, Andrew Regenscheid, and Angela Robinson. Digital Signature Standard (DSS). Federal Inf. Process. Stds. (NIST FIPS), National Institute of Standards and Technology, Gaithersburg, MD, USA, 2023. DOI: 10.6028/NIST.FIPS.186-5.

[CMW24] Giulio Corsi, Bill Marino, and Willow Wong. The spread of synthetic media on x. *Harvard Kennedy School Misinformation Review*, 2024. DOI: 10.37016/mr-2020-140.

[CNR+22] Matteo Campanelli, Anca Nitulescu, Carla Ràfols, Alexandros Zacharakis, and Arantxa Zapico. Linear-map vector commitments and their practical applications. In *Advances in Cryptology - ASIACRYPT 2022*, volume 13794 of *Lecture Notes in Computer Science*, pages 189–219. Springer, 2022. DOI: 10.1007/978-3-031-22972-5_7.

[Cod15] Mark Coddington. Clarifying journalism's quantitative turn: A typology for evaluating data journalism, computational journalism, and computer-assisted reporting. *Digital Journalism*, 3(3):331–348, 2015. DOI: 10.1080/21670811.2014.976400.

[Cur22] Monique Curet. CNN tweeted about "brave children" in Ukraine signing up to fight Russia. 2022. URL: https://www.politifact.com/factchecks/2022/apr/19/instagram-posts/cnn-did-not-tweet-about-children-ukraine-signing-f/ (visited on 03/15/2023).

[DB23] Trisha Datta and Dan Boneh. Using ZK proofs to fight disinformation. https://medium.com/@boneh/using-zk-proofs-to-fight-disinformation-17e7d57fe52f, 2023. (Visited on 11/28/2023).

[DCB25] Trisha Datta, Binyi Chen, and Dan Boneh. VerITAS: verifying image transformations at scale. In *IEEE Symposium on Security and Privacy - S&P 2025*. IEEE Computer Society, 2025. DOI: 10.1109/SP61157.2025.00097.

[DEH25] Stefan Dziembowski, Shahriar Ebrahimi, and Parisa Hassanizadeh. VIMz: verifiable image manipulation using folding-based zkSNARKs. *Proc. Priv. Enhancing Technol.*, 2025(2), 2025. URL: https://eprint.iacr.org/2024/1063.

[DGMS00] Premkumar T. Devanbu, Michael Gertz, Charles U. Martel, and Stuart G. Stubblebine. Authentic third-party data publication. In *Data and Application Security - IFIP 2000*, volume 201 of *IFIP Conference Proceedings*, pages 101–112. Kluwer, 2000. DOI: 10.1007/0-306-47008-X_9.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976. DOI: 10.1109/TIT.1976.1055638.

[DHC20] Ling Du, Anthony T. S. Ho, and Runmin Cong. Perceptual hashing for image authentication: A survey. *Signal Processing: Image Communication*, 81:115713, 2020. DOI: 10.1016/J.IMAGE.2019.115713.

[Dia22] Nicholas Diakopoulos. Predictive journalism: On the role of computational prospection in news media, 2022. URL: https://www.cjr.org/tow_center_reports/predictive-journalism-on-the-role-of-computational-prospection-in-news-media.php/ (visited on 03/15/2023).

[DKL+18] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Transactions of Cryptographic Hardware and Embedded Systems*, 2018(1):238–268, 2018. DOI: 10.13154/TCHES.V2018.I1.238-268.

[dMPPS14] Hermann de Meer, Henrich Christopher Pöhls, Joachim Posegga, and Kai Samelin. On the relation between redactable and sanitizable signature schemes. In *Engineering Secure Software and Systems - ESSoS 2014*, volume 8364 of *Lecture Notes in Computer Science*, pages 113–130. Springer, 2014. DOI: 10.1007/978-3-319-04897-0_8.

[Dom] Roney Domingos. É fake que g1 publicou reportagem afirmando que lula disse que, se eleito, irá revogar o pix. URL: https://g1.globo.com/fato-ou-fake/eleicoes/noticia/2022/10/06/e-fake-que-g1-publicou-reportagem-afirmando-que-lula-disse-

que-se-eleito-ira-revogar-o-pix.ghtml (visited on 03/06/2023).

[DPD+21] Viorela Dan, Britt Paris, Joan Donovan, Michael Hameleers, Jon Roozenbeek, Sander van der Linden, and Christian von Sikorski. Visual mis- and disinformation, social media, and democracy. *Journalism & Mass Communication Quarterly*, 98(3):641–664, 2021. DOI: 10.1177/10776990211035395.

[DSÁ20] Caitlin Drummond, Michael Siegrist, and Joseph Árvai. Limited effects of exposure to fake news about climate change. *Environmental Research Communications*, 2(8), 2020. DOI: 10.1088/2515-7620/abae77. Article 081003.

[Dwo15] Morris J. Dworkin. SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. Federal Inf. Process. Stds. (NIST FIPS), National Institute of Standards and Technology, Gaithersburg, MD, USA, 2015. DOI: 10.6028/NIST.FIPS.202.

[EEG25] Johanna Eggers, Simon Erfurth, and Marília Gehrke. Image authenticity in the age of AI: digital signatures as a defense against visual disinformation, 2025. Submitted to Cambrdige Disinformation Summit 2025.

[EFC24] EFCSN. European fact-checking standards network, 2024. URL: https://efcsn.com/projects/ (visited on 12/12/2024).

[EH23] Ziv Epstein and Aaron Hertzmann. Art and the science of generative ai. *Science*, 380(6650):1110–1111, 2023. DOI: 10.1126/science.adh4451.

[Erf24] Simon Erfurth. Digital signatures for authenticating compressed JPEG images. In *Security-Centric Strategies for Combating Information Disorder - SCID 2024*, page 4. ACM, 2024. DOI: 10.1145/3660512.3665522.

[Fag24] Evelyn Fagundes. É falsa imagem viral de homem resgatando bebê em rua alagada no rs, 2024. URL: https://lupa.uol.com.br/jornalismo/2024/05/21/e-falsa-imagem-viral-de-homem-resgatando-bebe-em-rua-alagada-no-rs (visited on 12/12/2024).

[FHK+20] Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. FALCON: Fast-Fourier lattice-based compact signatures over NTRU. Submission to the NIST Post-Quantum Project, v.1.2, 2020. https://falcon-sign.info/falcon.pdf.

[FKNP24] Giacomo Fenzi, Christian Knabenhans, Ngoc Khanh Nguyen, and Duc Tu Pham. Lova: lattice-based folding scheme from unstructured lattices. In *Advances in Cryptology - ASIACRYPT 2024*, volume 15487 of *Lecture Notes in Computer Science*, pages 303–326. Springer, 2024. DOI: 10.1007/978-981-96-0894-2_10.

[FN24] Richard Fletcher and Rasmus Kleis Nielsen. What does the public in six countries think of generative AI in news? Technical report, 2024. DOI: 10.60625/RISJ-4ZB8-CG87.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: practical solutions to identification and signature problems. In *Advances in Cryptology - CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986. DOI: 10.1007/3-540-47721-7_12.

[FS90] Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In *Symposium on Theory of Computing - STOC'90*, pages 416–426. ACM, 1990. DOI: 10.1145/100216.100272.

[GB21] Marília Gehrke and Marcia Benetti. Disinformation in brazil during the covid-19 pandemic: topics, platforms, and actors. *Fronteiras-Estudos Midiáticos*, 23(2):14–28, 2021. DOI: 10.4013/fem.2021.232.02.

[GDB04] Gunne Grankvist, Ulf Dahlstrand, and Anders Biel. The impact of environmental labelling on consumer preference: negative vs. positive labels. *Journal of Consumer Policy*, 27(2):213–230, 2004. DOI: 10.1023/b:copo.0000028167.54739.94.

[GE20] Kiran Garimella and Dean Eckles. Images and misinformation in political groups: evidence from whatsapp in india. *Harvard Kennedy School Misinformation Review*, 2020. DOI: 10.37016/mr-2020-030.

[GE23] Marília Gehrke and Simon Erfurth. Adding quotable signatures to the transparency repertoire in data journalism. In *Joint Computation+Journalism Symposium and European Data & Computational Journalism Conference 2023*, 2023. URL: https://www.datajconf.com/papers/CJ_DataJConf_2023_paper_17.pdf.

[GEdVH24] Marília Gehrke, Johanna Eggers, Claes de Vreese, and David Hopmann. What makes news (seem) authentic on social media? indicators from a qualitative study of young adults. *Digital Journalism*, 2024. DOI: 10.1080/21670811.2024.2399622.

[Geh20] Marília Gehrke. Transparency as a key element of data journalism: perceptions of brazilian professionals. In *Computation + Journalism Symposium conference proceedings*, 2020. URL: https://hdl.handle.net/11370/e7145ff5-8787-4243-8e24-f5b99f844cc7.

[Geh22] Marília Gehrke. *Os elementos de transparência no Jornalismo Guiado por Dados*. Insular, 2022. URL: https://insular.com.br/produto/os-elementos-de-transparencia-no-jornalismo-guiado-por-dados/.

[Geh23] Marília Gehrke. A decolonial feminist approach to gendered disinformation. In *Multidisciplinary International Symposium on Disinformation in Online Open Media - MISDOOM 2023*, 2023. URL: https://event.cwi.nl/misdoom-2023/abstracts.pdf.

[GGPR13] Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct NIZKs without PCPs. In *Advances in Cryptology - EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 626–645. Springer, 2013. DOI: 10.1007/978-3-642-38348-9_37.

[GM17] Marília Gehrke and Luciana Mielniczuk. Philip Meyer, the outsider who created Precision Journalism. *Intexto*, (39):4, 2017. DOI: 10.19132/1807-8583201739.4-13.

[GM24] Albert Garreta and Ignacio Manzur. FLI: folding lookup instances, 2024. DOI: 10.1007/978-981-96-0935-2_13.

[GPM+20] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. DOI: 10.1145/3422622.

[Gra16] Lucas Graves. *Deciding what's true: The rise of political fact-checking in American journalism*. Columbia University Press, 2016. DOI: 10.7312/grav17506.

[Gro16] Jens Groth. On the size of pairing-based non-interactive arguments. In *Advances in Cryptology - EUROCRYPT 2016*, volume 9666 of *Lecture Notes in Computer Science*, pages 305–326. Springer, 2016. DOI: 10.1007/978-3-662-49896-5_11.

[GW11] Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In *Symposium on Theory of Computing - STOC'11*, pages 99–108. ACM, 2011. DOI: 10.1145/1993636.1993651.

[GWC19] Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. PLONK: permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. *IACR Cryptol. ePrint Arch.*:953, 2019. URL: https://eprint.iacr.org/2019/953.

[Ham24] Michael Hameleers. The nature of visual disinformation online: a qualitative content analysis of alternative and social media in the netherlands. *Political Communication*:1–19, 2024. DOI: 10.1080/10584609.2024.2354389.

[Hau24] Liv Hausken. Photorealism versus photography. ai-generated depiction in the age of visual disinformation. *Journal of Aesthetics & Culture*, 16(1), 2024. DOI: 10.1080/20004214.2024.2340787.

[HF] Emily Heil and Paul Farhi. Fake editions of the washington post handed out at multiple locations in d.c. URL: https://www.washingtonpost.com/lifestyle/style/fake-editions-of-the-washington-post-handed-out-at-multiple-locations-in-dc/2019/01/16/1d6a0402-19a5-11e9-88fe-f9f77a3bcb6c_story.html (visited on 03/06/2023).

[HG13] Ryan Henry and Ian Goldberg. Batch proofs of partial knowledge. In *Applied Cryptography and Network Security - ACNS 2013*, volume 7954 of *Lecture Notes in Computer Science*, pages 502–517. Springer, 2013. DOI: 10.1007/978-3-642-38980-1_32.

[HHZ16] Stuart Haber, William G. Horne, and Miaomiao Zhang. Efficient transparent redactable signatures with a single signature invocation. *IACR Cryptol. ePrint Arch.*:1165, 2016. URL: http://eprint.iacr.org/2016/1165.

[HK13] Shoichi Hirose and Hidenori Kuwakado. Redactable signature scheme for tree-structured data based on merkle tree. In *Conference on Security and Cryptography - SECRYPT 2013*, pages 313–320. IEEE, 2013. URL: https://ieeexplore.ieee.org/document/7223180/.

[HKS24] Anne Hamby, Hongmin Kim, and Francesca Spezzano. Sensational stories: the role of narrative characteristics in distinguishing real and fake news and predicting their spread. *Journal of Business Research*, 170:114289, 2024. DOI: 10.1016/j.jbusres.2023.114289.

[HP24] Daniel Greneaa Hansen and Andreas Søndergaard Petersen. Kunstig intelligens er kendissvindlernes nyeste træk, 2024. URL: https://www.tjekdet.dk/indsigt/kunstig-intelligens-er-kendissvindlernes-nyeste-traek (visited on 12/12/2024).

[HRS16] Andreas Hülsing, Joost Rijneveld, and Fang Song. Mitigating multi-target attacks in hash-based signatures. In *Public-Key Cryptography - PKC 2016*, volume 9614 of *LECTURE NOTES IN COMPUTER SCIENCE*, pages 387–416. Springer, 2016. DOI: 10.1007/978-3-662-49384-7_15.

[HS10] Lynn Hunt and Vanessa R. Schwartz. Capturing the moment: images and eyewitnessing in history. *Journal of Visual Culture*, 9(3):259–271, 2010. DOI: 10.1177/1470412910380348. URL: http://dx.doi.org/10.1177/1470412910380348.

[HvdMV24] Michael Hameleers, Toni van der Meer, and Rens Vliegenthart. How persuasive are political cheapfakes disseminated via social media? the effects of out-of-context visual disinformation on message credibility and issue agreement. *Information, Communication & Society*:1–18, 2024. DOI: 10.1080/1369118x.2024.2388079.

[Ima24] Getty Images. Generate new ai images or modify our creative imagery, 2024. URL: https://www.gettyimages.dk/ai (visited on 12/12/2024).

[Int92] International Telecommunication Union. T.81 – Digital Compression and Coding of Continuous-Tone Still Images – Requirements and Guidelines. https://www.w3.org/Graphics/JPEG/itu-t81.pdf, 1992.

[IP18] Cherilyn Ireton and Julie Posetti. *Journalism,'Fake News'& Disinformation: Handbook for Journalism Education and Training*. UNESCO Publishing, 2018. URL: https://digitallibrary.un.org/record/1641987.

[Jen24] Magnus Stenaa Jensen. Detecting AI-manipulated content is a challenging arms race. https://www.dtu.dk/english/news/all-news/detecting-ai-manipulated-content-is-a-challenging-arms-race?id=f7be2a68-b6c4-4e39-bd21-6650009d287b, 2024. (Visited on 03/25/2024).

[JMSW02] Robert Johnson, David Molnar, Dawn Xiaodong Song, and David A. Wagner. Homomorphic signature schemes. In *Topics in Cryptology - CT-RSA 2002*, volume 2271 of *Lecture Notes in Computer Science*, pages 244–262. Springer, 2002. DOI: 10.1007/3-540-45760-7_17.

[JS21] Kirsten A. Johnson and Burton St. John III. Transparency in the news: the impact of self-disclosure and process disclosure on the perceived credibility of the journalist, the story, and the organization. *Journalism Studies*, 22(7):953–970, 2021. DOI: 10.1080/1461670X.2021.1910542.

[JWL11] Rob Johnson, Leif Walsh, and Michael Lamb. Homomorphic signatures for digital photographs. In *Financial Cryptography and Data Security - FC 2011*, volume 7035 of *Lecture Notes in Computer Science*, pages 141–157. Springer, 2011. DOI: 10.1007/978-3-642-27576-0_12.

[Kar10] Michael Karlsson. Rituals of transparency: evaluating online news outlets' uses of transparency rituals in the united states, united kingdom and sweden. *Journalism Studies*, 11(4):535–545, 2010. DOI: 10.1080/14616701003638400.

[Kar22] Michael Karlsson. *Transparency and journalism: a critical appraisal of a disruptive norm*. Disruptions: studies in digital journalism. Routledge, 2022. DOI: 10.4324/9780429340642.

[KFL23] Timo K. Koch, Lena Frischlich, and Eva Lermer. Effects of fact-checking warning labels and social endorsement cues on climate change fake news credibility and engagement on social media. *Journal of Applied Social Psychology*, 53(6):495–507, 2023. DOI: 10.1111/jasp.12959.

[KFM04] Maxwell N. Krohn, Michael J. Freedman, and David Mazières. On-the-fly verification of rateless erasure codes for efficient content distribution. In *IEEE Symposium on Security and Privacy - S&P 2004*, pages 226–240. IEEE Computer Society, 2004. DOI: 10.1109/SECPRI.2004.1301326.

[KFN18] Antonis Kalogeropoulos, Richard Fletcher, and Rasmus Kleis Nielsen. News brand attribution in distributed environments: do people know where they get their news? *New Media & Society*, 21(3):583–601, 2018. DOI: 10.1177/1461444481880.

[Kjæ24] Jakob Sorgenfri Kjær. Kunstig intelligens har lavet et af disse billeder, men hvilket?: kampen mod ai indeholder et kæmpe paradoks, 2024. URL: https://politiken.dk/danmark/art9794828/ (visited on 12/12/2024).

[KNSS19] Michael Kreutzer, Ruben Niederhagen, Kris Shrishak, and Hervais Simo Fhom. Quotable signatures using merkle trees. In *INFORMATIK 2019*, volume P-294 of *Lecture Notes in Informatik*, pages 473–477, 2019. DOI: 10.18420/INF2019_64.

[Kor17] Pawel Korus. Digital image integrity - a survey of protection and verification techniques. *Digit. Signal Process.*, 71:1–26, 2017. DOI: 10.1016/J.DSP.2017.08.009.

[Kot24] Abhiram Kothapalli. *A Theory of Composition for Proofs of Knowledge*. PhD thesis, Carnegie Mellon University, 2024. URL: https://www.andrew.cmu.edu/user/bparno/papers/kothapalli_thesis.pdf.

[KP23] Abhiram Kothapalli and Bryan Parno. Algebraic reductions of knowledge. In *Advances in Cryptology - CRYPTO 2023*, volume 14084 of *Lecture Notes in Computer Science*, pages 669–701. Springer, 2023. DOI: 10.1007/978-3-031-38551-3_21.

[Kri] Nikolaj Rodkjær Kristensen. Schmeichel om fabrikeret, mavesur udtalelse: "det er noget sludder". URL: https://www.tjekdet.dk/faktatjek/schmeichel-om-fabrikeret-mavesur-udtalelse-det-er-noget-sludder (visited on 03/06/2023).

[KS22] Abhiram Kothapalli and Srinath T. V. Setty. SuperNova: proving universal machine executions without universal circuits. *IACR Cryptol. ePrint Arch.*:1758, 2022. URL: https://eprint.iacr.org/2022/1758.

[KS24a] Abhiram Kothapalli and Srinath T. V. Setty. HyperNova: recursive arguments for customizable constraint systems. In *Advances in Cryptology - CRYPTO 2024*, volume 14929 of *Lecture Notes in Computer Science*, pages 345–379. Springer, 2024. DOI: 10.1007/978-3-031-68403-6_11.

[KS24b] Abhiram Kothapalli and Srinath T. V. Setty. NeutronNova: folding everything that reduces to zero-check. *IACR Cryptol. ePrint Arch.*:1606, 2024. URL: https://eprint.iacr.org/2024/1606.

[KST22] Abhiram Kothapalli, Srinath T. V. Setty, and Ioanna Tzialla. Nova: recursive zero-knowledge arguments from folding schemes. In *Advances in Cryptology - CRYPTO 2022*, volume 13510 of *Lecture Notes in Computer Science*, pages 359–388. Springer, 2022. DOI: 10.1007/978-3-031-15985-5_13.

[KZG10] Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In *Advances in Cryptology - ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 177–194. Springer, 2010. DOI: 10.1007/978-3-642-17373-8_11.

[LBB+18] David M. J. Lazer, Matthew A. Baum, Yochai Benkler, Adam J. Berinsky, Kelly M. Greenhill, Filippo Menczer, Miriam J. Metzger, Brendan Nyhan, Gordon Pennycook, David Rothschild, Michael Schudson, Steven A. Sloman, Cass R. Sunstein, Emily A. Thorson, Duncan J. Watts, and Jonathan L. Zittrain. The science of fake news. *Science*, 359(6380):1094–1096, 2018. DOI: 10.1126/science.aao2998.

[LC01] Ching-Yung Lin and Shih-Fu Chang. A robust image authentication method distinguishing JPEG compression from malicious manipulation. *IEEE Transactions on Circuits and Systems for Video Technology - (TCSVT)*, 11(2):153–168, 2001. DOI: 10.1109/76.905982.

[LDG+22] Zhan Liu, Matthieu Delaloye, Nicole Glassey Balet, Sébastien Hersant, Frédéric Gris, and Laurent Sciboz. Trust in the news: a digital labelling solution for journalistic contents. *Online journal of communication and media technologies*, 12, 2022. DOI: 10.30935/ojcmt/11528.

[LEC+24] Stephan Lewandowsky, Ullrich K. H. Ecker, John Cook, Sander van der Linden, Jon Roozenbeek, Naomi Oreskes, and Lee C. McIntyre. Liars know they are lying: differentiating disinformation from disagreement. *Humanities and Social Sciences Communications*, 11(1), 2024. DOI: 10.1057/s41599-024-03503-6.

[LES+12] Stephan Lewandowsky, Ullrich K. H. Ecker, Colleen M. Seifert, Norbert Schwarz, and John Cook. Misinformation and its correction: continued influence and successful debiasing. *Psychological Science in the Public Interest*, 13(3):106–131, 2012. DOI: 10.1177/1529100612451018.

[Lil24] Jordan Liles. Video shows a young kamala harris dancing on 'soul train'? 2024. URL: https://www.snopes.com/fact-check/kamala-harris-dancing-on-soul-train/ (visited on 12/12/2024).

[LLK13] Ben Laurie, Adam Langley, and Emilia Käsper. Certificate transparency. *RFC*, 6962:1–27, 2013. URL: https://www.rfc-editor.org/rfc/rfc6962.

[LP24] Diane Leblanc-Albarel and Bart Preneel. Black-box collision attacks on the neuralhash perceptual hash function. *IACR Cryptol. ePrint Arch.*:1869, 2024. URL: https://eprint.iacr.org/2024/1869.

[LVG12] Yao-Chung Lin, David P. Varodayan, and Bernd Girod. Image authentication using distributed source coding. *IEEE Trans. Image Process.*, 21(1):273–283, 2012. DOI: 10.1109/TIP.2011.2157515.

[Lyo23] Santiago Lyon. Leica launches world's first camera with content credentials. https://contentauthenticity.org/blog/leica-launches-worlds-first-camera-with-content-credentials, 2023. (Visited on 11/28/2023).

[Mer80] Ralph C. Merkle. Protocols for public key cryptosystems. In *IEEE Symposium on Security and Privacy - S&P 1980*, pages 122–134. IEEE Computer Society, 1980. DOI: 10.1109/SP.1980.10006.

[Mer89] Ralph C. Merkle. A certified digital signature. In *Advances in Cryptology - CRYPTO '89*, volume 435, pages 218–238. Springer, 1989. DOI: 10.1007/0-387-34805-0_21.

[Mey02] Philip Meyer. *Precision journalism: a reporter's introduction to social science methods*. Rowman & Littlefield Publishers, 4th edition, 2002.

[MH80] David Marr and Ellen Hildreth. Theory of edge detection. *Proceedings of the Royal Society B*, 207(1167):187–217, 1980. DOI: 10.1098/rspb.1980.0020.

[MR02] Silvio Micali and Ronald L. Rivest. Transitive signature schemes. In *Topics in Cryptology - CT-RSA 2002*, volume 2271 of *Lecture Notes in Computer Science*, pages 236–243. Springer, 2002. DOI: 10.1007/3-540-45760-7_16.

[MSLL21] Maria D. Molina, S. Shyam Sundar, Thai Le, and Dongwon Lee. "fake news" is not simply false information: a concept explication and taxonomy of online content. *American Behavioral Scientist*, 65(2):180–212, 2021. DOI: 10.1177/0002764219878.

[MVVZ25] Pierpaolo Della Monica, Ivan Visconti, Andrea Vitaletti, and Marco Zecchini. Trust nobody: privacy-preserving proofs for edited photos with your laptop. In *IEEE Symposium on Security and Privacy - S&P 2025*. IEEE Computer Society, 2025. DOI: 10.1109/SP61157.2025.00014.

[Nat24a] National Institute of Standards and Technology. Module-Lattice-Based Digital Signature Standard. Federal Inf. Process. Stds. (NIST FIPS), National Institute of Standards and Technology, Gaithersburg, MD, USA, 2024. DOI: 10.6028/NIST.FIPS.204.

[Nat24b] National Institute of Standards and Technology. Stateless Hash-Based Digital Signature Standard. Federal Inf. Process. Stds. (NIST FIPS), National Institute of Standards and Technology, Gaithersburg, MD, USA, 2024. DOI: 10.6028/NIST.FIPS.205.

[NDC+24] Wilson D. Nguyen, Trisha Datta, Binyi Chen, Nirvan Tyagi, and Dan Boneh. Mangrove: a scalable framework for folding-based SNARKs. In *Advances in Cryptology - CRYPTO 2024*, volume 14929 of *Lecture Notes in Computer Science*, pages 308–344. Springer, 2024. DOI: 10.1007/978-3-031-68403-6_10.

[NFKN19] Nic Newman, Richard Fletcher, Antonis Kalogeropoulos, and Rasmus Kleis Nielsen. Reuters Institute Digital News Report 2019. Technical report, Reuters Institute for the Study of Journalism, 2019. URL: https://www.digitalnewsreport.org/survey/2019/.

[NFR+22] Nic Newman, Richard Fletcher, Craig T. Robertson, Kirsten Eddy, and Rasmus Kleis Nielsen. Reuters Institute Digital News Report 2022. Technical report, Reuters Institute for the Study of Journalism, 2022. URL: https://reutersinstitute.politics.ox.ac.uk/sites/default/files/2022-06/Digital_News-Report_2022.pdf.

[NFR+24] Nic Newman, Richard Fletcher, Craig T. Robertson, Amy Ross Arguedas, and Rasmus Kleis Nielsen. Reuters Institute Digital News Report 2024. Technical report, Reuters Institute for the Study of Journalism, 2024. URL: https://reutersinstitute.politics.ox.ac.uk/digital-news-report/2024.

[NNY+24] David Na, Samuel Nathanson, Yungjun Yoo, Yinzhi Cao, and Lanier Watkins. Showcasing the threat of scalable generative ai disinformation through social media simulation. In pages 1–2. IEEE, 2024. DOI: 10.1109/infocomwkshps61880.2024.10620878.

[NT16] Assa Naveh and Eran Tromer. PhotoProof: cryptographic image authentication for any set of permissible transformations. In *IEEE Symposium on Security and Privacy - S&P 2016*, pages 255–271. IEEE Computer Society, 2016. DOI: 10.1109/SP.2016.23.

[Nys24] Annabelle Nyst. History of chatgpt: a timeline of the meteoric rise of generative ai chatbots, 2024. URL: https://www.searchenginejournal.com/history-of-chatgpt-timeline/488370/ (visited on 12/02/2024).

[OH24] Lisa O'Carroll and Alex Hern. Eu calls on tech firms to outline plans to tackle deepfakes amid election fears, 2024. URL: https://www.theguardian.com/business/2024/mar/14/eu-calls-on-tech-firms-outline-plans-to-tackle-deepfakes-amid-election-fears-google-facebook-x (visited on 12/12/2024).

[OLRW20] Katherine Ognyanova, David Lazer, Ronald E. Robertson, and Christo Wilson. Misinformation in action: fake news exposure is linked to lower trust in media, higher trust in government when your side is in power. *Misinformation Review*, 1(4), 2020. DOI: 10.37016/mr-2020-024.

[PBN+23] Sejin Paik, Sarah Bonna, Ekaterina Novozhilova, Ge Gao, Jongin Kim, Derry Wijaya, and Margrit Betke. The affective nature of ai-generated news images: impact on visual journalism. In pages 1–8. IEEE, 2023. DOI: 10.1109/acii59096.2023.10388166.

[PC86] Richard E. Petty and John T. Cacioppo. The elaboration likelihood model of persuasion. In *Advances in Experimental Social Psychology*, pages 123–205. Elsevier, 1986. DOI: 10.1016/s0065-2601(08)60214-2.

[PD19] Britt Paris and Joan Donovan. Deepfakes and cheap fakes: fakes: the manipulation of audio and visual evidence, 2019. URL: https://datasociety.net/library/deepfakes-and-cheap-fakes/.

[Ped91] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology - CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer, 1991. DOI: 10.1007/3-540-46766-1_9.

[PLS23] Yilang Peng, Yingdan Lu, and Cuihua Shen. An agenda for studying credibility perceptions of visual misinformation. *Political Communication*, 40(2):225–237, 2023. DOI: 10.1080/10584609.2023.2175398.

[PLZ+09] Nikolay Ponomarenko, Vladimir Lukin, Alexander Zelensky, Karen Egiazarian, Marco Carli, and Federica Battisti. TID2008 — A database for evaluation of full-reference visual quality assessment metrics. *Advances of Modern Radioelectronics*, 10(4):30–45, 2009. URL: https://www.ponomarenko.info/papers/mre2009tid.pdf.

[PM18] Julie Posetti and Alice Matthews. A short guide to the history of 'fake news' and disinformation. *International Center for Journalists*, 2018.

[PNC24] Nihal Poredi, Deeraj Nagothu, and Yu Chen. Authenticating ai-generated social media images using frequency domain analysis. In pages 534–539. IEEE, 2024. DOI: 10.1109/ccnc51664.2024.10454640.

[PPSS23] Corina Pelau, Mihai-Ionut Pop, Mihaela Stanescu, and Grigorie Sanda. The breaking news effect and its impact on the credibility and trust in information posted on social media. *Electronics*, 12(2):423, 2023. DOI: 10.3390/electronics12020423.

[Reu] Reuters Fact Check. Fact check-screenshot of bbc news report on russia is fake. URL: https://www.reuters.com/article/factcheck-bbc-screenshotfalse-idUSL2N2VL1D4 (visited on 03/06/2023).

[RN23] Carlos Diaz Ruiz and Tomas Nilsson. Disinformation and echo chambers: how disinformation circulates on social media through identity-driven controversies. *Journal of Public Policy & Marketing*, 42(1):18–35, 2023. DOI: 10.1177/07439156221103852.

[Rog13] Simon Rogers. *Facts are sacred: the power of data*. Faber and Faber, Guardian Books, 2013.

[Rüs06] Jörn Rüsen, editor. Meaning and representation in history, 2006. DOI: 10.3167/9781571817761. URL: http://dx.doi.org/10.3167/9781571817761.

[RZ23] Carla Ràfols and Alexandros Zacharakis. Folding schemes with selective verification. In *Progress in Cryptology - LATINCRYPT 2023*, volume 14168 of *Lecture Notes in Computer Science*, pages 229–248. Springer, 2023. DOI: 10.1007/978-3-031-44469-2_12.

[SBV+13] Srinath T. V. Setty, Benjamin Braun, Victor Vu, Andrew J. Blumberg, Bryan Parno, and Michael Walfish. Resolving the conflict between generality and plausibility in verified computation. In *EuroSys '13*, pages 71–84. ACM, 2013. DOI: 10.1145/2465351.2465359.

[SBZ01] Ron Steinfeld, Laurence Bull, and Yuliang Zheng. Content extraction signatures. In *Information Security and Cryptology - ICISC 2001*, volume 2288 of *Lecture Notes in Computer Science*, pages 285–304. Springer, 2001. DOI: 10.1007/3-540-45861-1_22.

[Sch] Arthur H. Schiochet. É falso que instituto alemão apontou fraude nas eleições do brasil. URL: https://lupa.uol.com.br/jornalismo/2022/11/17/e-falso-que-instituto-alemao-apontou-fraude-nas-eleicoes-do-brasil (visited on 03/06/2023).

[Sch15] A. Brad Schwartz. The infamous "war of the worlds" radio broadcast was a magnificent fluke, 2015. URL: https://www.smithsonianmag.com/history/infamous-war-worlds-radio-broadcast-was-magnificent-fluke-180955180/ (visited on 11/28/2024).

[Sch22] Arthur Schiochet. É falso que instituto alemão apontou fraude nas eleições do Brasil, 2022. URL: https://lupa.uol.com.br/jornalismo/2022/11/17/e-falso-que-instituto-alemao-apontou-fraude-nas-eleicoes-do-brasil (visited on 03/15/2023).

[Set20] Srinath T. V. Setty. Spartan: efficient and general-purpose zksnarks without trusted setup. In *Advances in Cryptology - CRYPTO 2020*, volume 12172 of *Lecture Notes in Computer Science*, pages 704–737. Springer, 2020. DOI: 10.1007/978-3-030-56877-1_25.

[SF90] Laura A. Sanchis and Mark A. Fulk. On the efficient generation of language instances. *SIAM Journal on Computing*, 19(2):281–296, 1990. DOI: 10.1137/0219019.

[SIM+22] Emily Sidnam-Mauch, Bernat Ivancsics, Ayana Monroe, Eve Washington, Errol Francis II, Kelly Caine, Joseph Bonneau, and Susan E. McGregor. Usable cryptographic provenance: A proactive complement to fact-checking for mitigating misinformation. In *AAAI Conference on Web and Social Media - ICWSM 2022*, 2022. DOI: 10.36190/2022.55.

[SK20] Leonie Schaewitz and Nicole C. Krämer. Combating disinformation: effects of timing and correction format on factual knowledge and personal beliefs. In *Multidisciplinary International Symposium on Disinformation in Open Online Media - MISDOOM 2020*, volume 12259 of *Lecture Notes in Computer Science*, pages 233–245. Springer, 2020. DOI: 10.1007/978-3-030-61841-4_16.

[SPB+12] Kai Samelin, Henrich Christopher Pöhls, Arne Bilzhause, Joachim Posegga, and Hermann de Meer. On structural signatures for tree data structures. In *Applied Cryptography and Network Security - ACNS 2012*, volume 7341 of *Lecture Notes in Computer Science*, pages 171–187. Springer, 2012. DOI: 10.1007/978-3-642-31284-7_11.

[SR21] Felipe Soares and Raquel Recuero. How the mainstream media help to spread disinformation about covid-19. *M/C Journal*, 24(1), 2021. DOI: 10.5204/mcj.2735.

[STW23] Srinath T. V. Setty, Justin Thaler, and Riad S. Wahby. Customizable constraint systems for succinct arguments. *IACR Cryptol. ePrint Arch.*:552, 2023. URL: https://eprint.iacr.org/2023/552.

[Sun98] S. Shyam Sundar. Effect of source attribution on perception of online news stories. *Journalism & Mass Communication Quarterly*, 75(1):55–68, 1998. DOI: 10.1177/107769909807500108.

[TAD+20] T.J. Thomson, Daniel Angus, Paula Dootson, Edward Hurcombe, and Adam Smith. Visual mis/disinformation in journalism and public communications: current verification practices, challenges, and future opportunities. *Journalism Practice*, 16(5):938–962, 2020. DOI: 10.1080/17512786.2020.1832139.

[tAI 24] Jim the AI Whisperer. Artificial intelligence and "what ifs": prince william welcomes prince harry home at king charles' coronation! 2024. URL: https://medium.com/@JimTheAIWhisperer/prince-william-welcomes-prince-harry-home-at-king-charles-coronation-d84673a42432 (visited on 12/12/2024).

[TB14] Milan Tuba and Nebojsa Bacanin. JPEG quantization tables selection by the firefly algorithm. In *Conference on Multimedia Computing and Systems - ICMCS 2014*, pages 153–158. IEEE, 2014. DOI: 10.1109/ICMCS.2014.6911315.

[TDB16] Giulia Traverso, Denise Demirel, and Johannes Buchmann. *Homomorphic Signature Schemes - A Survey*. Springer Briefs in Computer Science. Springer, 2016. DOI: 10.1007/978-3-319-32115-8.

[The22] The Independent JPEG Group (IJG). JPEG Standard Reference Implementation (version 9e). https://jpegclub.org/reference/reference-sources/, 2022.

[Tho00] Brian Thornton. The moon hoax: debates about ethics in 1835 new york newspapers. *Journal of Mass Media Ethics*, 15(2):89–100, 2000. DOI: 10.1207/S15327728JMME1502_3.

[Thy23] Rose Marie Pontoppidan Thyssen. Ritzau-direktør stoler på, at fotografer ikke snyder med kunstig intelligens, 2023. URL: https://journalisten.dk/ritzau-direktoer-stoler-paa-at-fotografer-ikke-snyder-med-kunstig-intelligens/ (visited on 12/12/2024).

[Ton23] Jingrong Tong. *Data for journalism: between transparency and accountability*. Disruptions. Routledge, Taylor & Francis Group, 2023. DOI: 10.4324/9781003030089.

[TSGS19] Terry Traylor, Jerey Straub, Gurmeet, and Nicholas Snell. Classifying fake news articles using natural language processing to identify in-article attribution as a supervised learning estimator. In *IEEE International Conference on Semantic Computing - ICSC 2019*, pages 445–449. IEEE, 2019. DOI: 10.1109/ICOSC.2019.8665593.

[TTM24] T. J. Thomson, Ryan J. Thomas, and Phoebe Matich. Generative visual ai in news organizations: challenges, opportunities, perceptions, and policies. *Digital Journalism*:1–22, 2024. DOI: 10.1080/21670811.2024.2331769.

[Uni19] International Telecommunication Union. Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks. Technical report, 2019. URL: www.itu.int/rec/T-REC-X.509-201910-I/en.

[US14] Juliane Urban and Wolfgang Schweiger. News quality from the recipients' perspective. *Journalism Studies*, 15(6):821–840, 2014. DOI: 10.1080/1461670X.2013.856670.

[Val08] Paul Valiant. Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In *Theory of Cryptography - TCC 2008*, volume 4948 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2008. DOI: 10.1007/978-3-540-78524-8_1.

[vdMHO23] Toni G. L. A. van der Meer, Michael Hameleers, and Jakob Ohme. Can fighting misinformation have a negative spillover effect? how warnings for the threat of misinformation can decrease general news credibility. *Journalism Studies*, 24(6):803–823, 2023. DOI: 10.1080/1461670x.2023.2187652.

[VHS20] Michail Vafeiadis, Jiangxue Ashley Han, and Fuyuan Shen. News storytelling through images: examining the effects of narratives and visuals in news coverage of issues. *International Journal of Communication*, 14:21, 2020. URL: https://ijoc.org/index.php/ijoc/article/view/12227/0.

[VRA18] Soroush Vosoughi, Deb Roy, and Sinan Aral. The spread of true and false news online. *Science*, 359(6380):1146–1151, 2018. DOI: 10.1126/science.aap9559.

[W3C21] W3C. Clipboard api and events, W3C working draft, 6 August 2021, 2021. URL: https://www.w3.org/TR/2021/WD-clipboard-apis-20210806/.

[WAA+24] Jim Woodcock, Mikkel Schmidt Andersen, Diego F. Aranha, Stefan Hallerstede, Simon Thrane Hansen, Nikolaj Kuhne Jakobsen, Tomas Kulik, Peter Gorm Larsen, Hugo Daniel Macedo, Carlos Ignacio Isasa Martin, and Victor Alexander Mtsimbe Norrild. State of the art report: verified computation, 2024. arXiv: 2308.15191 [cs.CR]. URL: https://arxiv.org/abs/2308.15191.

[Wal91] Gregory K. Wallace. The JPEG still picture compression standard. *Communications of the ACM*, 34(4):30–44, 1991. DOI: 10.1145/103085.103089.

[WCM24] Farah Al Wardani, Kaustuv Chaudhuri, and Dale Miller. About trust and proof: an experimental framework for heterogeneous verification. In *The Practice of Formal Methods: Essays in Honour of Cliff Jones*, volume 14781 of *Lecture Notes in Computer Science*, pages 162–183. Springer, 2024. DOI: 10.1007/978-3-031-66673-5_9.

[Web] Allan Weber. The USC-SIPI image database. https://sipi.usc.edu/database/database.php. (Visited on 04/04/2024).

[WGN24] Teresa Weikmann, Hannah Greber, and Alina Nikolaou. After deception: how falling for a deepfake affects the way we see, hear, and experience media. *The International Journal of Press/Politics*, 2024. DOI: 10.1177/19401612241233539.

[WL22] Teresa Weikmann and Sophie Lecheler. Visual disinformation in a digital age: a literature synthesis and research agenda. *New Media & Society*, 25(12):3696–3713, 2022. DOI: 10.1177/14614448221141648.

[WSB03] Zhou Wang, Eero P. Simoncelli, and Alan C. Bovik. Multiscale structural similarity for image quality assessment. In *IEEE Asilomar Conference on Signals, Systems and Computers*, volume 2, pages 1398–1402. IEEE, 2003. DOI: 10.1109/ACSSC.2003.1292216.

[YLF+22] Thomas Yurek, Licheng Luo, Jaiden Fairoze, Aniket Kate, and Andrew Miller. hbACSS: how to robustly share many secrets. In *Network and Distributed System Security Symposium - NDSS 2022*. The Internet Society, 2022. URL: https://www.ndss-symposium.org/ndss-paper/auto-draft-245/.

[YW22] Kang Yang and Xiao Wang. Non-interactive zero-knowledge proofs to multiple verifiers. In *Advances in Cryptology - ASIACRYPT 2022*, volume 13793 of *Lecture Notes in Computer Science*, pages 517–546. Springer, 2022. DOI: 10.1007/978-3-031-22969-5_18.

[ZCF24] Hadas Zeilberger, Binyi Chen, and Ben Fisch. BaseFold: efficient field-agnostic polynomial commitment schemes from foldable codes. In *Advances in Cryptology - CRYPTO 2024*, volume 14929 of *Lecture Notes in Computer Science*, pages 138–169. Springer, 2024. DOI: 10.1007/978-3-031-68403-6_5.

[ZFC19] Koosha Zarei, Reza Farahbakhsh, and Noel Crespi. Deep dive on politician impersonating accounts in social media. In pages 1–6. IEEE, 2019. DOI: 10.1109/iscc47284.2019.8969645.

[ZKMH07] Fang Zhao, Ton Kalker, Muriel Médard, and Keesook J. Han. Signatures for content distribution with network coding. In *IEEE Symposium on Information Theory - ISIT 2007*, pages 556–560. IEEE, 2007. DOI: 10.1109/ISIT.2007.4557283.

[ZSL04] Bin Benjamin Zhu, Mitchell D. Swanson, and Shipeng Li. Encryption and authentication for scalable multimedia: current state of the art and challenges. *Internet Multimedia Management Systems V*, 5601:157–170, 2004. DOI: 10.1117/12.571869.

[ZXH+22] Jiaheng Zhang, Tiancheng Xie, Thang Hoang, Elaine Shi, and Yupeng Zhang. Polynomial commitment with a one-to-many prover and applications. In *USENIX Security 2022*, pages 2965–2982. USENIX Association, 2022. URL: https://www.usenix.org/conference/usenixsecurity22/presentation/zhang-jiaheng.

[ZYO+24] Chengru Zhang, Xiao Yang, David Oswald, Mark Ryan, and Philipp Jovanovic. Eva: efficient ivc-based authentication of lossy-encoded videos. *IACR Cryptol. ePrint Arch.*:1436, 2024. URL: https://eprint.iacr.org/2024/1436.

[ZZMZ11a] Lin Zhang, Lei Zhang, Xuanqin Mou, and David Zhang. Feature SIMilarity index for IQA. https://web.comp.polyu.edu.hk/cslzhang/IQA/FSIM/FSIM.htm, 2011.

[ZZMZ11b] Lin Zhang, Lei Zhang, Xuanqin Mou, and David Zhang. FSIM: A feature similarity index for image quality assessment. *IEEE Trans. Image Process.*, 20(8):2378–2386, 2011. DOI: 10.1109/TIP.2011.2109730.