

Improved Rate for Non-Malleable Codes and Time-Lock Puzzles

Cody Freitag  

Northeastern University, Boston, MA, USA
Hebrew University of Jerusalem, Israel

Ilan Komargodski  

Hebrew University of Jerusalem, Israel

Manu Kondapaneni  

Northeastern University, Boston, MS, USA

Jad Silbak  

Massachusetts Institute of Technology, Boston, MA, USA

Abstract

Non-malleable codes allow a sender to transmit a message to a receiver, while providing a “best-possible” integrity guarantee to ensure that no attacker – who cannot already decode the message – can meaningfully tamper the message in transit. If tampered, the received message should either be invalid or unrelated to the original message. Non-malleable time-lock puzzles (TLPs) are a special case of non-malleable codes for bounded polynomial-depth tampering with very efficient encoding.

In this work, we give generic techniques for constructing non-malleable codes and non-malleable TLPs with improved *rate*, which captures the ratio of a message’s length to its encoding length.

A key contribution of our work is identifying a security notion for non-malleability, which we term “CCA-hiding”, sufficient for our compilers. CCA-hiding is a relaxation of CCA-security for encryption or commitments to the fine-grained setting of codes, and requires that the encoded message remains hidden, even given a decoding oracle for any other codeword. Intriguingly, CCA-hiding does not imply non-malleability in the fine-grained setting, as is the case for encryption and commitments.

Using our new techniques, we give the following constructions:

- Rate-1 CCA-hiding TLPs in the plain model.
- Rate-1 non-malleable codes for bounded polynomial-depth tampering in the auxiliary-input random oracle model (AI-ROM).
- Rate-(1/2) non-malleable TLPs in the AI-ROM.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography

Keywords and phrases Non-malleable codes, Time-lock puzzles

Digital Object Identifier 10.4230/LIPIcs.ITCS.2026.62

Funding *Cody Freitag*: Supported in part by a Khoury College Distinguished Postdoctoral Fellowship and the European Union (ERC, SCALE, 101162665). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

Ilan Komargodski: Supported in part by the European Union (ERC, SCALE, 101162665). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

Jad Silbak: Supported in part by a Khoury College Distinguished Postdoctoral Fellowship. Part of this work was done while the author was a postdoctoral research fellow at the Simons Institute for the Theory of Computing (research supported in part by a grant from the UC Noyce Initiative to the Simons Institute for the Theory of Computing.)



© Cody Freitag, Ilan Komargodski, Manu Kondapaneni, and Jad Silbak;
licensed under Creative Commons License CC-BY 4.0

17th Innovations in Theoretical Computer Science Conference (ITCS 2026).

Editor: Shubhangi Saraf; Article No. 62; pp. 62:1–62:24

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Acknowledgements We thank Jesko Dujmovic for helpful discussion on the paper and for first pointing out the uninstantiability result for our construction in the random oracle model.

1 Introduction

Non-malleable codes (NMCs), introduced by Dziembowski, Pietrzak, and Wichs [35], allow a sender to encode a message while maintaining a “best-possible” integrity guarantee for the underlying message. Namely, it should be impossible to tamper – or *maul* – a codeword in transit to one that decodes to another related message. This security notion is only achievable against bounded tampering adversaries that cannot decode the message, as otherwise such an adversary could decode the message and simply output a codeword for a related message. Non-malleable codes have been widely studied and have been constructed for a variety of adversarial tampering classes including split-state, bounded space, bounded depth, and bounded size tampering (see, e.g., [35, 67, 3, 38, 12, 28, 37, 4, 10, 14, 32, 11, 16, 9, 33, 15, 17]).

Of particular interest are *non-malleable time-lock puzzles* (TLPs), which are NMCs for bounded polynomial-depth tampering, where the encoding algorithm is required to be more efficient than decoding. In recent years, non-malleable TLPs have received significant interest due to their applications to distributed coin-flipping, auctions, and broadcast (see, e.g., [23, 40, 21, 79, 80, 8]).

In this work, we are interested in the efficiency of non-malleable codes and time-lock puzzles. A particularly important efficiency measure is a code’s *rate*, which is an asymptotic notion capturing the ratio of the message length to its encoding length. The “holy grail” in terms of efficiency is building rate-1 codes, where the codeword is essentially the same length as the underlying message (up to additive terms that depend only on the security parameter). There is also great practical interest in improving the rate of non-malleable TLPs for their aforementioned applications to the blockchain setting where communication and storage costs are often prohibitively expensive (see e.g., [1] for example costs for Ethereum). As it stands, most known NMCs (e.g., [3, 12, 28, 4, 10, 14, 11, 16, 9, 15]) and non-malleable time-lock puzzles (e.g., [56, 33, 40, 79]) are *not* rate-1.

If we only care about hiding the underlying message m , it is easy to construct a rate-1 encoding Enc given any encoding Enc' and a pseudo-random generator PRG: simply encode a PRG seed s used to mask the message with a one-time pad,

$$\text{Enc}(m) := (\text{Enc}'(s), \text{PRG}(s) \oplus m).$$

This same general idea works in many contexts including for encryption and commitments. Note, however, that this construction is clearly malleable; adding 1 to the one-time pad results in an encoding of $m \oplus 1$. Indeed, non-malleability is notoriously difficult to compose. For example, $\text{Enc}(a, b) := (\text{Enc}'(a), \text{Enc}'(b))$ is malleable even if Enc' is non-malleable (e.g., a tampering adversary could simply swap the two encodings). So, in this paper, we ask:

*Can we generically improve the rate of any
non-malleable code or time-lock puzzle?*

1.1 Our Results

We answer the above question affirmatively by giving various transformations and constructions improving the rate of existing non-malleable codes and non-malleable TLPs in different settings. A key contribution of our work is identifying the right security notion for the underlying codes in our transformations. We call this notion *chosen-code attack* (CCA) hiding.

Non-malleability vs. CCA-hiding

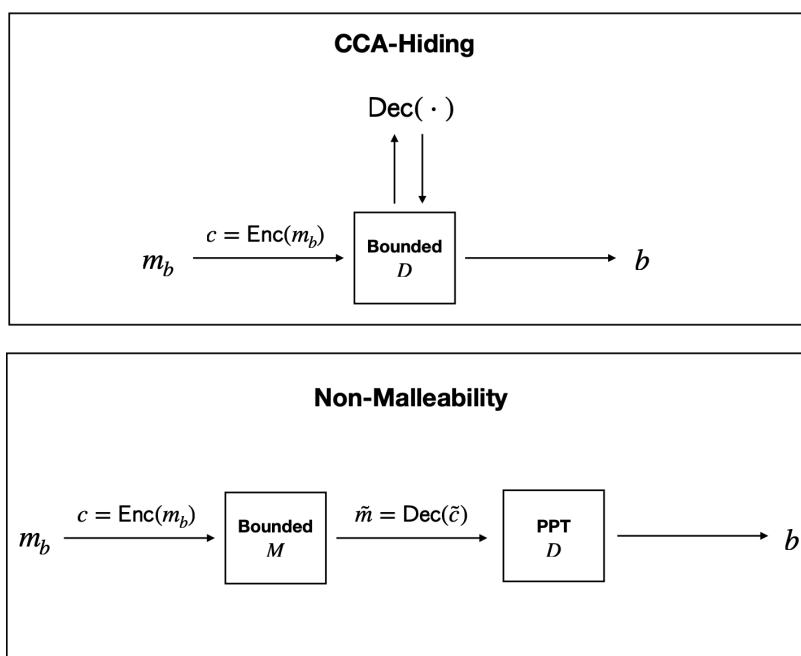
We are interested in encoding schemes consisting of public, polynomial-time encoding and decoding algorithms (Enc, Dec). While conceptually similar, we compare CCA-hiding and non-malleability explicitly to clarify their differences.

- *CCA-hiding* requires that, for any pair of messages m_0, m_1 , any *bounded* distinguisher D (from a specified class of adversaries that can't compute $\text{Dec}(\cdot)$) can't predict b given a code $c \leftarrow \text{Enc}(m_b)$, even given an oracle for $\text{Dec}(\cdot)$ on any input $c' \neq c$.
- *Non-malleability* is modeled via the following “meddler-in-the-middle” (MIM) experiment for a *bounded* mauling adversary M and message m ,

$$\text{MIM}_M(m) := \text{Dec}(M(\text{Enc}(m))),$$

where $\text{Dec}(\cdot)$ returns \perp if M simply copies the encoding c it receives as input. Non-malleability requires that, for any pair of messages m_0, m_1 , any polynomial-time distinguisher D (that *can* compute $\text{Dec}(\cdot)$) can't predict b given $\tilde{m} \leftarrow \text{MIM}_M(m_b)$.¹

We summarize the above two security games in Figure 1 below.



■ **Figure 1** Comparison of CCA-hiding and non-malleability security games for polynomial-time encoding and decoding algorithms (Enc, Dec). In the games above, $\text{Dec}(\cdot)$ outputs \perp on input the challenge encoding c , and we refer to an algorithm as “bounded” if it comes from a specified class that cannot compute $\text{Dec}(\cdot)$.

The above two definitions are similar in that they give the security experiment limited access to a decoding oracle $\text{Dec}(\cdot)$. However, they differ in that CCA-hiding ensures indistinguishability against bounded adversaries that can adaptively query $\text{Dec}(\cdot)$, whereas non-malleability ensures indistinguishability against *arbitrary polynomial-time* adversaries for the output of one call to $\text{Dec}(\cdot)$. While the classical notion of CCA-security implies

¹ We follow the notion of non-malleability from [40] for non-malleable TLPs, which is necessary for their applications to multi-party coin-flipping and auctions. In the NMC literature, this is sometimes referred to as “strong” non-malleability where we additionally protect against attacks that maul the code c to a fresh encoding $c' \neq c$ for the same underlying message m .

non-malleability for encryption or commitments, we will see that this is not the case in the fine-grained setting of codes.² Still, we will show that CCA-hiding is a useful tool for composition, and in particular, for achieving non-malleability.

Summary of our results

In the rest of the section, we give an overview of our main results, summarized as follows:

- We give a generic transformation from any CCA-hiding encoding scheme to a rate-1 CCA-hiding encoding scheme. This transformation relies only on the existence of collision resistant hash functions. We note that this transformation also applies to non-interactive commitments.
- We next give a direct construction of a rate-1 non-malleable code for bounded polynomial-depth tampering. This construction is in the auxiliary-input random oracle model (AI-ROM) of Unruh [81], where the adversary in the security game is allowed to depend arbitrarily on the random oracle.
- Lastly, we give a generic transformation from any CCA-hiding encoding scheme to a rate-1/2 *non-malleable* encoding scheme in the AI-ROM.

1.1.1 Rate-1 compiler for CCA-hiding

Our first result is a generic rate-1 compiler for CCA-hiding codes.

► **Theorem 1.1** (Rate-1 Compiler for CCA-Hiding; Informal, see Theorem 3.6). *Assuming collision-resistant hash functions, there is a generic compiler that given a CCA-hiding encoding scheme (with any rate), outputs a rate-1 CCA-hiding encoding scheme.*

We additionally show that the non-malleable TLPs from [33, 40] satisfy CCA-hiding (see the full version for details). We prove the construction of [33] satisfies CCA-hiding in the plain model from the same assumptions of TLPs, keyless multi-collision resistant hash functions (MCRH), injective one-way functions (OWF), and non-interactive witness indistinguishable (NIWI) proofs, all with sub-exponential security. The other construction of [40] is proven secure assuming the existence of sub-exponentially secure TLPs in the AI-ROM. We prove their construction satisfies CCA-hiding assuming only polynomially secure TLPs in the AI-ROM. As a result, we get the following corollaries from our generic compiler of Theorem 1.1.

► **Corollary 1.2** (Rate-1 CCA-Secure Time-Lock Puzzles). *There exist rate-1 CCA-hiding time-lock puzzles*

- *in the plain model assuming TLPs, keyless MCRH, injective OWFs, and NIWI proofs, all with sub-exponential security, and*
- *in the AI-ROM assuming TLPs with polynomial security.*

Applications to non-malleable non-interactive commitments

Our compilers work for arbitrary encoding schemes (Enc, Dec) with security against some class of polynomially bounded adversaries. While we usually think of Dec(\cdot) as being relatively efficient, non-interactive commitments can be viewed as encoding schemes with inefficient decoding algorithms; you can simply guess the commitment randomness. This makes it possible to achieve notions of non-malleability against arbitrary polynomial-time tampering.

² In particular, [40] give a construction that *is* malleable yet satisfies a restricted notion of CCA-hiding for one oracle call to Dec(\cdot).

Theorem 1.1 gives a simple transformation to build rate-1 CCA-secure (and hence non-malleable) non-interactive commitments from any CCA-secure non-interactive commitment. Somewhat surprisingly, we observe that the recent work of [43] directly gives a rate-1 CCA-secure non-interactive commitment, even though they don't explicitly make note of this.³ Our compiler gives a generic way to achieve rate-1 for any future CCA-secure non-interactive commitment from simpler assumptions, or for the construction of [22] (assuming it is also CCA-secure).

1.1.2 Rate-1 non-malleable codes in the AI-ROM

One could hope the codes based on Theorem 1.1 are already non-malleable. Unfortunately, we show an explicit attack on non-malleability for these constructions (see Section 2.1.1). As such, we require new techniques for achieving non-malleable codes or time-lock puzzles in the fine-grained setting.

For our next result, we give a direct construction of a rate-1 NMC for bounded polynomial-depth tampering. This construction relies on a random oracle and security is proven in the AI-ROM.

► **Theorem 1.3** (Rate-1 Non-Malleable Code in the AI-ROM; Informal, see Theorem 4.8). *There exists a rate-1 non-malleable code for bounded polynomial-depth tampering in the AI-ROM.*

In contrast to time-lock puzzles, we emphasize that encoding in the construction above requires polynomial depth which is larger than the depth-bound for the tampering class. In a similar setting, the recent work of Ball, Shaltiel, and Silbak [17] gives *rate-1* NMCs for bounded polynomial-size tampering. Their constructions only tolerate computational indistinguishability up to a fixed inverse polynomial, which is inherent based on the complexity-theoretic style assumptions they use. In contrast, our construction above achieves negligible security.

We also emphasize that our construction of Theorem 1.3 requires only a random oracle and makes no additional assumptions. Mahmoody, Moran, and Vadhan [68] show that this is only possible when encoding is as slow as decoding, so TLPs are impossible to achieve in this setting.

1.1.3 Rate-1/2 compiler for non-malleability from CCA-hiding in the AI-ROM

We next give a generic compiler for CCA-hiding codes that gives rate-1/2 non-malleable codes in the AI-ROM.

► **Theorem 1.4** (Rate-1/2 Compiler from CCA-Hiding to Non-Malleability in the AI-ROM; Informal, see Theorem 5.11). *In the AI-ROM, there is a generic compiler that given a CCA-hiding encoding scheme, outputs a rate-1/2 non-malleable encoding scheme.*

Again, plugging in the AI-ROM TLP of [40] into this new compiler, we get the following direct improvement over their non-malleable TLP as an immediate corollary to Theorem 1.4.

► **Corollary 1.5** (Rate-1/2 Non-Malleable TLPs in the AI-ROM). *Assuming polynomially-secure TLPs, there exists a rate-1/2 non-malleable time-lock puzzle in the AI-ROM.*

³ Their primary motivation was to remove the non-black-box use of crypto primitives in the previous constructions of [65, 22].

Applications to multi-party coin-flipping and auctions

In the full Theorem 5.11, we prove that the compiler of Theorem 1.4 above results in a stronger notion of functional non-malleability introduced by [40] as a way to bypass the impossibility of concurrent non-malleability for TLPs. At a high level, functional non-malleability with respect to a multi-input function f captures the following scenario:

- A bounded tampering adversary receives $c \leftarrow \text{Enc}(m)$ and outputs a sequence of codes $\tilde{c}_1, \dots, \tilde{c}_k$ that decode to values $\tilde{m}_1, \dots, \tilde{m}_k$ (ignoring any copied codes). The adversary wins if $f(\tilde{m}_1, \dots, \tilde{m}_k)$ is non-trivially related to m .

The work of [40] show that this is useful for applications to distributed coin-flipping and auction protocols. Using our non-malleable TLPs in these protocols results in significantly less overall communication while relying on weaker assumptions. To highlight this, we briefly outline the application to fair coin-flipping, and refer to their paper for further details.

To generate ℓ bits of shared randomness, each party i uses a non-malleable TLP to “commit” to a string $r_i \in \{0, 1\}^\ell$ within a fixed window of time where the time-locked security holds. Each puzzle is then opened to reveal r_i – potentially by calling $\text{Dec}(\cdot)$ to ensure all TLPs are opened – and then all parties agree on the random string $r = \oplus_i r_i$. Now, suppose that party j is honest and commits to a random value $r_j \leftarrow \{0, 1\}^\ell$. An attacker who wants to bias the output r must come up with TLPs to values $\{r_i\}_{i \neq j}$ within the short window of time where TLPs are accepted. Functional non-malleability for the \oplus function guarantees that $\oplus_{i \neq j} r_i$ doesn’t depend on the value of r_j , so it follows that the output r is indistinguishable from a uniformly random string.

On random oracle uninstantiability results for TLPs

While the compiler above for TLPs is provably secure in the AI-ROM, we show that if the underlying CCA-hiding TLP satisfies certain strong “fully homomorphic” properties, the resulting TLP is actually malleable for any concrete instantiation of the random oracle. Still, we believe that this transformation is secure for most natural underlying CCA-hiding TLPs. Moreover, constructions in the AI-ROM serve as useful heuristics for practical constructions and are important first steps towards constructions in the plain model.

This random oracle uninstantiability result is similar to other such results [26] where the scheme is only computationally secure in the ROM, as is the case for the Fujisaki-Okamoto transform [41, 40, 25, 45] and the Fiat-Shamir heuristic for arguments [39, 18, 44, 20, 57]. Namely, for certain choice of the underlying computational ingredients, there are similar attacks for any instantiation of the random oracle.

We give a formal treatment for our uninstantiability result for the resulting TLP from Theorem 1.4 in Section 5.1. As our rate-1 NMC from Theorem 1.3 is proven unconditionally secure in the AI-ROM, we are unaware of any similar attacks on this construction.

Paper Organization

We discuss related work in Section 1.2 and give a brief comparison with the most related constructions in the full version. In Section 2, we give an overview for the main techniques used in the analysis of our constructions above. To state our formal results, we give necessary preliminaries and definitions in the full version. In this abbreviated version, we describe our formal construction of our CCA-hiding compiler in Section 3. We give our non-malleability compiler from CCA-hiding in Section 5. We then give our direct non-malleable code construction in Section 4. All associated proofs are deferred to the full version. We additionally give the constructions and proofs of CCA-hiding for the TLP constructions of [33, 40] in the AI-ROM and plain model in the full version.

1.2 Related Work

Non-malleable encoding schemes including codes, time-lock puzzles, and non-interactive commitments have received significant interest since the seminal motivating works of [70, 76, 34]. We provide a brief overview of the related literature in these areas.

Non-malleable codes

Non-malleable codes were first introduced by [35] and have been constructed for a variety of adversarial tampering classes including split-state tampering [35, 67, 28, 4, 3], algebraic tampering [9], bounded space tampering [37, 14], bounded size tampering [35, 38, 14, 11, 15, 17], or bounded depth tampering [12, 10, 16, 33]. The various works above give constructions in a variety of settings (probabilistic vs. explicit constructions, CRS vs. plain model, inverse polynomial vs. negligible security), but in this work, we are primarily interested in explicit constructions without a CRS that achieve negligible security against either bounded polynomial-size or ℓ -depth adversaries. Also, to the best of our knowledge, CCA-hiding has not been explicitly studied in the context of non-malleable codes.

Non-malleable time-lock puzzles

Non-malleable TLPs (and the related notion of non-malleable non-interactive timed commitments) have received significant attention due to their applications to distributed protocols like coin-flipping, fair auctions, and broadcast [23, 40, 21, 79, 80, 36, 8]. While some non-malleable TLP constructions rely on adversary-independent setup [56, 21, 79, 80, 29, 7], we are primarily interested in the fully non-interactive setting in this work following [40] and inspired by the original notion of TLPs from [77]. We note that Katz, Loss, and Xu [56] define a notion of CCA-hiding for non-interactive timed commitments, which was subsequently considered in the works of [79, 29, 7]. [79] show that this notion suffices for “short-lived” applications that only require indistinguishability for a limited amount of time. Following [40], we are primarily concerned with applications that guarantee the standard notion of computational indistinguishability against arbitrary polynomial-time attackers.

Non-malleable non-interactive commitments

Non-malleable and CCA-secure (even interactive) commitments have been extremely useful for building more efficient and secure multi-party computation protocols [19, 82, 27, 64, 61, 48, 65, 24]. This has led to a long sequence of works trying to minimize the round complexity and assumptions needed for constructing such commitments [34, 19, 74, 73, 66, 71, 62, 75, 82, 46, 63, 47, 49, 30, 31, 58, 65, 60, 22, 53, 50, 42, 59, 43]. Despite known black-box barriers for constructing non-interactive non-malleable commitments [72], there has been significant progress recently in the fully non-interactive setting that we consider in this work [71, 65, 22, 53, 42, 59, 43].

Rate compilers for other non-malleable primitives

Our work is not the first to explore rate-optimizing compilers for non-malleable codes (NMCs). All existing compilers, including ours, follow a similar overarching design inspired by the “hybrid” encryption paradigm: each combines a low-rate non-malleable code with a form of symmetric-key authenticated encryption. However, despite this high-level resemblance, the specific instantiations of these components differ significantly across works, and each setting presents unique challenges.

The most closely related work to ours is [17] as they consider tampering classes that may touch the entire codeword. However, as discussed before, their construction does not achieve negligible security. All other compilers are for settings where the tampering is subject to some form of locality constraint: each tampered output bit is computed by a function that has access only to partial information from the original codeword.

- **Local Tampering:** The first rate compiler by [6] boosted NMCs for 1-local tampering to rate 1 by combining threshold secret sharing with bit-flipping and error correction. Later, [52] extended these ideas (together with ideas from [13]) to achieving rate 1 NMCs for $c \log n$ -local functions.
- **Computational Split-State Tampering:** [2] showed how to obtain rate 1 NMCs for polynomial-time split-state tampering by encoding a high-rate authenticated encryption key using an augmented split-state NMC, introducing the notion of augmented non-malleability to address joint tampering across components.
- **Split-State Tampering:** A sequence of works [54, 55, 51, 5] used compilers to improve the rate of NMCs for split-state and t -state tampering, culminating in an explicit rate $1/3$ NMC for 2-state tampering – the best known to date. These compilers rely on leakage-resilient encodings combined with information-theoretic MACs and use seeded extractors and low-rate NMCs to protect the seeds and keys. The key technical challenge lies in preserving independence between tampering states.

2 Technical Overview

For the sake of this overview, we focus on the setting of TLPs, but we note that we provide results for general codes when relevant in the technical sections of the paper. We start by recalling the definition of a TLP. Throughout this overview, we consider a fixed security parameter λ and time bound t for simplicity.

A TLP consists of an encoding algorithm Enc and a decoding algorithm Dec . Correctness for TLPs stipulates that if you encode a message m resulting in a code c , decoding c should give back m . In short, $m = \text{Dec}(\text{Enc}(m))$. We recall the two security notions of CCA-hiding and non-malleability from Figure 1 specifically for the setting of TLPs:

- *CCA-hiding* requires that, for any pair of equal-length messages m_0, m_1 , any distinguishing adversary D running in depth $\ll t$ can't predict b given a code $c \leftarrow \text{Enc}(m_b)$, even given an oracle for $\text{Dec}(\cdot)$ on any input $c' \neq c$.
- *Non-malleability* is modelled via the following “meddler-in-the-middle” (MIM) experiment for a depth $\ll t$ mauling adversary M and message m ,

$$\text{MIM}_M(m) := \text{Dec}(M(\text{Enc}(m))),$$

where $\text{Dec}(\cdot)$ returns \perp if M simply copies its input. Non-malleability requires that, for any pair of equal-length messages m_0, m_1 , any polynomial-time distinguisher D (that may run in depth $\geq t$) can't predict b given $\tilde{m} \leftarrow \text{MIM}_M(m_b)$.

In Section 2.1, we describe a generic compiler that converts any CCA-hiding TLP into a rate-1 CCA-hiding TLP (Theorem 1.1). We then describe why this falls short of giving a non-malleable TLP. In Section 2.2, we give a generic compiler in the AI-ROM that gives a rate-1/2 non-malleable TLP from any CCA-hiding TLP (Theorem 1.4). Finally, in Section 2.3, we give a direct construction of a rate-1 non-malleable code in the AI-ROM (Theorem 1.3).

2.1 Rate-1 CCA-Hiding Compiler

Our overall goal is to give a generic compiler that converts any “sufficiently non-malleable” TLP $(\text{Enc}', \text{Dec}')$ into a new non-malleable TLP (Enc, Dec) with improved rate. Towards this goal, we start with a simple construction that gives a rate-1 TLP. The idea is to simply encode a short seed s for a PRG G and then use $G(s)$ to mask the message m . An encoding $\text{Enc}(m)$ would consist of an inner encoding $c' \leftarrow \text{Enc}'(s)$ with a one-time pad of the message $\text{pad} = G(s) \oplus m$. While this does *hide* the message m , it is clearly malleable. Given a TLP $c = (c', \text{pad}) \leftarrow \text{Enc}(m)$, a mauling attack that outputs $(c', \text{pad} \oplus 1)$ results in a valid TLP for $m \oplus 1$.

To circumvent this simple attack, we use a collision-resistant hash to bind the underlying TLP to the message m . Namely, together with s , you also encode a hash key hk and a hash of m . This gives our first compiler,

$$\text{Enc}(m) := (c' \leftarrow \text{Enc}'(s, \text{hk}, \text{Hash}(\text{hk}, m)), \text{pad} = G(s) \oplus m).$$

To decode, you can first decode $(s, \text{hk}, \text{dig})$ from $\text{Dec}'(c')$, unmask pad to get $\hat{m} = G(s) \oplus \text{pad}$, and check that $\text{dig} = \text{Hash}(\text{hk}, \hat{m})$. In case the check fails, the decoding algorithm will simply output \perp .

The construction is clearly rate-1 since the overhead is simply the underlying TLP encoding with length independent of $|m|$. Proving security, on the other hand, turns out to be a bit subtle. In particular, we identify that the underlying TLP $(\text{Enc}', \text{Dec}')$ needs to be CCA-hiding, and we will show that (Enc, Dec) is then also CCA-hiding.

Security analysis

While we eventually argue CCA-hiding of this construction, it will be instructive to present and analyze the security as if it were a non-malleability experiment for a bounded depth distinguisher D . For messages m_0, m_1 , recall that the experiment consists of a bounded depth mauler M that receives a challenge TLP $c \leftarrow \text{Enc}(m_b)$ for a random bit b as input. M tries to maul c to a new TLP \tilde{c} , with underlying message \tilde{m} . Then, a distinguisher D given \tilde{m} tries to predict the original bit b .

We now look at the different ways that M could potentially maul the challenge TLP c . Recall that the encoding c consists of two parts: $c' \leftarrow \text{Enc}'(s, \text{hk}, \text{dig})$ for $\text{dig} = \text{Hash}(\text{hk}, m_b)$, and $\text{pad} = G(s) \oplus m_b$. M then outputs a new TLP $\tilde{c} = (\tilde{c}', \tilde{\text{pad}})$.

- First, if $\tilde{c}' = c'$, meaning that M copied the first input, then it must be the case that $\tilde{\text{pad}} \neq \text{pad}$ (otherwise the whole TLP was copied, which isn't allowed). Since \tilde{c}' was copied, it decodes to the same underlying values $(s, \text{hk}, \text{dig})$ as c' . The candidate message $\hat{m} = G(s) \oplus \tilde{\text{pad}}$ must be different from m_b since $\tilde{\text{pad}} \neq \text{pad}$. But that means that $\text{Dec}(\tilde{c})$ outputs a non- \perp value (that could potentially be useful for D) only if $\text{Hash}(\text{hk}, \hat{m}) = \text{dig} = \text{Hash}(\text{hk}, m_b)$, resulting in a collision!
- If $\tilde{c}' \neq c'$, then the underlying values $(\tilde{s}, \tilde{\text{hk}}, \tilde{\text{dig}})$ should be “unrelated” to $(s, \text{hk}, \text{dig})$ – and hence m_b – if the underlying TLP is non-malleable. So, we would hope to show that this means $\tilde{m} = G(\tilde{s}) \oplus \tilde{\text{pad}}$ is also unrelated to m_b . Formalizing this intuition is a bit tricky, and it turns out that non-malleability alone will not suffice for the underlying TLP.

To analyze the second case above, let's examine the non-malleability game in a bit more detail.

- Non-Malleability Experiment for (Enc, Dec) :
 1. A code $c = (c', \text{pad}) \leftarrow \text{Enc}(m_b)$ is sampled for m_b .
 - Recall that $c' \leftarrow \text{Enc}'(s, \text{hk}, \text{dig})$ for $\text{dig} = \text{Hash}(\text{hk}, m_b)$ and $\text{pad} = G(s) \oplus m_b$.

2. The mauler outputs a tampered codeword $\tilde{c} = (c', \tilde{\text{pad}}) \leftarrow M(c)$.
3. The MIM experiment outputs $\tilde{m} = \text{Dec}(\tilde{c})$ as long as $\tilde{c} \neq c$.
 - Recall that $\text{Dec}(c', \tilde{\text{pad}})$ decodes $(\tilde{s}, \tilde{\text{hk}}, \tilde{\text{dig}}) = \text{Dec}'(c')$, computes a candidate message $\hat{m} = \text{G}(\tilde{s}) \oplus \tilde{\text{pad}}$, and outputs $\tilde{m} = \hat{m}$ if $\tilde{\text{dig}} = \text{Hash}(\tilde{\text{hk}}, \hat{m})$.
 - If $c' = c$, we argued that this results in a collision. So, it must be the case that $\tilde{c} \neq c$.
4. The distinguisher predicts a bit $b' = D(\tilde{m})$ and wins if $b' = b$.

To formalize the intuition that \tilde{m} should be “unrelated” to $(s, \text{hk}, \text{dig})$, we need to argue that the output of the non-malleability experiment doesn’t significantly change in a hybrid experiment where we instead sample the value $c' \leftarrow \text{Enc}'(0^{|s|+|\text{hk}|+|\text{dig}|})$. This requires that the non-malleability experiment is “hiding” for Enc' , but this can’t be the case since the experiment has to compute $\text{Dec}(\tilde{c})$! This is where CCA-hiding comes to save the day; it guarantees that hiding of Enc' still holds even if you can compute $\text{Dec}'(\tilde{c})$ on $\tilde{c} \neq c$. Furthermore, we’ve already argued this is the only interesting case based on the collision-resistance of Hash .

Now, after making c' independent of s and m_b , we can use the pseudo-randomness of G to move to a hybrid where $\text{pad} \leftarrow \{0, 1\}^{|\text{pad}|}$ is uniformly random and independent of m_b . The resulting experiment is independent of b , so D has no advantage for predicting b .

We note that we can extend the above argument to also capture the setting where M might make additional adaptive queries to $\text{Dec}(\cdot)$, but we still require that the distinguisher D is depth bounded. This game corresponds exactly to the CCA-hiding security game, which is why the above construction works for general CCA-hiding.

2.1.1 An Explicit Mauling Attack

One might assume that the above TLP is already non-malleable, at least under some sufficient non-malleability property of the underlying TLP. On the contrary, this is not just an issue in the analysis; there is an explicit mauling attack! The only gap in the argument above is that we assumed the distinguisher D in the non-malleability experiment ran in bounded depth $\ll t$. So, any attack on non-malleability must leverage the additional power of a polynomial-time distinguisher D that can run in depth $\geq t$. At a very high level, we will do this by having the bounded depth mauler somehow “pass useful information” to D that can be used in an attack.

Recall that $\text{Enc}(m) := (c', \text{pad})$ for $c' \leftarrow \text{Enc}'(s, \text{hk}, \text{Hash}(\text{hk}, m))$. We observe that c' is much shorter than m , but a polynomial-time D can easily distinguish c' generated using m_0 versus m_1 . Namely, D can compute $(s, \text{hk}, \text{dig}) = \text{Dec}'(c')$ and check if dig is equal to either $\text{Hash}(\text{hk}, m_0)$ or $\text{Hash}(\text{hk}, m_1)$. To turn this into an actual mauling attack, note that a mauler M on input $c = (c', \text{pad})$ can “pass” c' to D by outputting $\tilde{c} = \text{Enc}(c')$ (padding c' to the appropriate length). By definition of the non-malleability experiment, the distinguisher D then receives c' as input, which it can easily distinguish as described above.

2.2 Non-Malleability from CCA-Hiding in the AI-ROM

We next give a compiler for any CCA-hiding TLP $(\text{Enc}', \text{Dec}')$ in the random oracle model that avoids all such mauling attacks, resulting in a non-malleable TLP (Enc, Dec) . In the random oracle model, we give all relevant parties (sender, receiver, and attackers) oracle access to a common random function \mathcal{O} . When proving security, we allow the mauler M and distinguisher D to depend *arbitrarily* on the random oracle, modelled via the auxiliary-input random oracle model (AI-ROM) due to Unruh [81]. This captures the realistic scenario

where we instantiate \mathcal{O} with a fixed function like SHA-3, and we give the attacker arbitrary pre-processing time on the function to compute an advice string for the online mauler and distinguisher.

The above mauling attack used the fact that a small part of the TLP can be distinguished in polynomial-time, so could be passed to D via an encoding. Our goal now is to try to rule out all such attacks in (Enc, Dec) . First, we want to make sure everything encoded in c' itself hides m against polynomial-time D . So, we conceptually replace the digest $\text{dig} = \text{Hash}(\text{hk}, m)$ with a succinct, hiding commitment to m (using \mathcal{O}). The second issue is that, given s , any bit of $\text{pad} = \text{G}(s) \oplus m$ can be unmasked leading to a mauling attack. We fix this by requiring “knowledge” of an additional random string r in order to unmask pad (using the extraction properties of \mathcal{O}). Intuitively, we will make r long enough to ensure that, information theoretically, it cannot be “passed” to D . Since the decoder will also need to know r , this will slightly increase the length of our encoding.

Construction

To encode a message m , we first encode a seed s with a hash key hk and a succinct, hiding commitment to m via $\mathcal{O}(\text{hk}||m||\rho)$ using randomness ρ . Next, we give out a random string $r \leftarrow \{0, 1\}^{|m|}$ that is the same length as m . Finally, we use $\mathcal{O}(s||r)$ to mask the message m along with the commitment randomness ρ . Altogether, our new construction is

$$\begin{aligned} \text{Enc}^{\mathcal{O}}(m) &:= (c' \leftarrow \text{Enc}'(s, \text{hk}, \mathcal{O}(\text{hk}||m||\rho)), r \leftarrow \{0, 1\}^{|m|+\lambda}, \\ &\quad \text{pad} = \mathcal{O}(s||r) \oplus (m||\rho)). \end{aligned}$$

To decode, you can first decode $(s, \text{hk}, \text{com})$ from c' , unmask pad to get a candidate message and randomness $(\hat{m}||\hat{\rho}) = \text{pad} \oplus \mathcal{O}(s||r)$, and output $m := \hat{m}$ if $\text{com} = \mathcal{O}(\text{hk}||\hat{m}||\hat{\rho})$. The construction is rate-1/2 since it includes r which is roughly the same length as the message m , and otherwise the additional overheads depend only on the security parameter λ and are independent of $|m|$.

Security analysis

Our goal is to rule out all possible mauling attacks. Let $c = (c', r, \text{pad})$ denote the challenge TLP, and $\tilde{c} = (\tilde{c}', \tilde{r}, \tilde{\text{pad}})$ denote the mauled TLP. Then every mauling attack must fall into one of the following two cases:

- First, if $(\tilde{c}', \tilde{r}) = (c', r)$, it again must be the case that $\tilde{\text{pad}} \neq \text{pad}$ if the whole TLP wasn't copied. But when c' and r are unchanged, the mauled TLP must use the same values s and r to unmask $\tilde{\text{pad}} \neq \text{pad}$. So, when decoding \tilde{c} , this results in different candidate message and randomness $(\hat{m}||\hat{\rho}) = \mathcal{O}(s||r) \oplus \tilde{\text{pad}}$ compared to the m and ρ used to generate the challenge TLP c . It follows that if \tilde{c} is a valid encoding, this results in a collision on $\mathcal{O}(\text{hk}||\cdot)$ since $\mathcal{O}(\text{hk}||m||\rho) = \text{com} = \mathcal{O}(\text{hk}||\hat{m}||\hat{\rho})$, which is unlikely to occur.
- Otherwise, it must be the case that either $\tilde{c}' \neq c'$ or $\tilde{r} \neq r$, meaning the value $\mathcal{O}(\tilde{s}||\tilde{r})$ used to mask the mauled message is likely to be different when decoding \tilde{c} compared to the challenge TLP c . Our goal in this case is to argue that the original value $\mathcal{O}(s||r)$ used to mask $(m_b||\rho)$ can be lazily sampled independent from the rest of the non-malleability experiment. If this were the case, $\text{pad} = \mathcal{O}(s||r) \oplus (m_b||\rho)$ would reveal no information about m_b or ρ , meaning that the commitment $\text{com} = \mathcal{O}(\text{hk}||m_b||\rho)$ also reveals no information about m_b , and we'd be done!

As stated above, our goal is to argue that the output of $\mathcal{O}(s||r)$ can be sampled lazily and is independent of the rest of the experiment, meaning the rest of the experiment never queries \mathcal{O} on $(s||r)$. To help argue this, let's take a closer look at how the non-malleability experiment queries \mathcal{O} on such values.

- Non-Malleability Experiment for $(\text{Enc}^\mathcal{O}, \text{Dec}^\mathcal{O})$:
 1. A code $c \leftarrow \text{Enc}^\mathcal{O}(m_b)$ is sampled for m_b .
 2. The mauler outputs a tampered codeword $\tilde{c} = (\tilde{c}', \tilde{r}, \tilde{\text{pad}}) \leftarrow M^\mathcal{O}(c)$.
 3. The MIM experiment outputs $\tilde{m} = \text{Dec}^\mathcal{O}(\tilde{c})$ as long as $\tilde{c} \neq c$.
 - If $\tilde{c}' \neq c'$, compute \tilde{m} by querying $\mathcal{O}(\tilde{s}||\tilde{r})$ for \tilde{s} output by $\text{Dec}'(\tilde{c}')$.
 - If $\tilde{c}' = c'$ but $\tilde{r} \neq r$, compute \tilde{m} by querying $\mathcal{O}(s||\tilde{r})$.
 4. The distinguisher predicts a bit $b' = D^\mathcal{O}(\tilde{m})$ and wins if $b' = b$.

To argue that the experiment doesn't query $\mathcal{O}(s||r)$ computed by $\text{Enc}^\mathcal{O}(m_b)$, we separately argue that $M^\mathcal{O}$, $\text{Dec}^\mathcal{O}$, and $D^\mathcal{O}$ are unlikely to query $\mathcal{O}(s||r)$. At a very high level, this is because

- $M^\mathcal{O}$ and $\text{Dec}^\mathcal{O}$ have incomplete information on s , and
- $D^\mathcal{O}$ has incomplete information on r .

To formalize the first point, we rely on the CCA-hiding of $(\text{Enc}', \text{Dec}')$. Namely, steps 2. and 3. in the case $\tilde{c}' \neq c'$ above can be computed in bounded depth $\ll t$ given the challenge TLP c using $M^\mathcal{O}$ and given oracle access to $\text{Dec}'(\tilde{c}')$. So, if M or Dec in this case ever query \mathcal{O} at some value $(s||\cdot)$ for the random s underlying c_{tlp} , this must violate CCA-hiding of $(\text{Enc}', \text{Dec}')$. For the case $\tilde{c}' = c'$, note that this step only ever queries $\mathcal{O}(s||\tilde{r})$ for $\tilde{r} \neq r$, so it also never queries $\mathcal{O}(s||r)$.

To formalize the second point, note that $D^\mathcal{O}$ only receives a string \tilde{m} of length $|m_b|$ as input and makes polynomially many queries to \mathcal{O} . This means that the set of all possible queries for which $D^\mathcal{O}$ might have some information on is at most $2^{|m_b|} \cdot \text{poly}(\lambda)$. However, since we chose $|r| = |m_b| + \lambda$, r has $|m_b| + \lambda$ bits of entropy, implying that $D^\mathcal{O}$ is very unlikely to make any query of the form $(\cdot||r)$. Note that this argument is information theoretic, and only makes use of the fact that the input to D is bounded, limiting the amount of information that can be passed to D . Indeed, this argument holds for any polynomial-time D that may run in depth $\geq t$.

As mentioned above, it follows that $\mathcal{O}(s||r)$ can be replaced with a uniformly random string in the non-malleability experiment, so pad is a uniformly random string independent of the rest of the experiment. Then, since $\text{com} = \mathcal{O}(\text{hk}||m_b||\rho)$ hides m for random ρ , the experiment is independent of m_b , so even a polynomial-time distinguisher D has no advantage for predicting b .

Extension to functional non-malleability

In Section 5, we additionally show that the construction above satisfies *functional* non-malleability, where the mauler may output a sequence of codes $\tilde{c}_1, \dots, \tilde{c}_k$ with underlying messages $\tilde{m}_1, \dots, \tilde{m}_k$. Then, we show that the output of any function $y = f(\tilde{m}_1, \dots, \tilde{m}_k)$ doesn't depend on the input message m to the experiment as long as $|r|$ is as long as the length of the functions output $|y|$. Furthermore, the analysis above holds for *any* (potentially unbounded) function f that doesn't query \mathcal{O} , or for bounded depth functions f that may query \mathcal{O} .

Random oracle uninstantiability

In Section 5.1, we show that if the underlying TLP in the compiler satisfies certain “fully homomorphic” properties, then the resulting TLP is malleable for any concrete instantiation H of the oracle \mathcal{O} .⁴ At a high level, this is because our argument above relies on the

⁴ Somewhat surprisingly, a TLP can satisfy both CCA-hiding while also being fully homomorphic as we define in Definition 5.14.

extractability of \mathcal{O} ; you have to “know” s to compute $\mathcal{O}(s||r)$. However, we show a concrete attack in case you can homomorphically evaluate $H(s||r)$ under the inner TLP for some concrete instantiation H for \mathcal{O} . Still, for most natural TLP candidates, the resulting TLP may still be non-malleable, and we also view this as an important first step towards a secure construction in the plain model.

2.3 Rate-1 Non-Malleable Code in the AI-ROM

We next describe a direct construction of a non-malleable code that achieves rate-1 in the AI-ROM compared to the rate-1/2 non-malleable TLP described in Section 2.2. Both constructions are non-malleable for the same class of bounded $\ll t$ polynomial-depth tampering, but the code has “slow” encoding that runs in depth $\geq t$ (recall TLPs require encoding time $\ll t$).

For starters, we observe that slow encoding alone already thwarts the mauling attack described in Section 2.1 above. In the attack, the mauling adversary M on input a challenge code c output a tampered code $\tilde{c} \leftarrow \text{Enc}(c')$ where c' was some small part of c that could be distinguished in polynomial-time. So, if the mauler can’t compute $\text{Enc}(c')$ in bounded depth, this particular attack isn’t possible. Indeed, the recent non-malleable code of [17] for bounded polynomial-size tampering achieves rate-1 by relying on a slow encoding function. However, their construction only achieves inverse polynomial indistinguishability, so it is natural to wonder if it is possible to build a “cryptographic” rate-1 non-malleable code that achieves negligible security.

Construction

The blueprint of our construction is similar to our previous compilers and relies on a random oracle \mathcal{O} . Rather than encode the seed s , however, we simply give s in the clear. We then use the t -fold iteration of \mathcal{O} on input s , which we denote by $\mathcal{O}^{(t)}(s)$, to mask the message m . We note that an encoding consisting of s and $\text{pad} = \mathcal{O}^{(t)}(s) \oplus m$ already suffices to hide m against bounded depth attackers, but is clearly malleable by just adding 1 to pad .

To avoid this simple attack, we again rely on a succinct, hiding commitment to bind the mauler to the seed s and message m . We also mask the commitment randomness ρ and append a commitment $\text{com} = \mathcal{O}(s||m||\rho)$ to the encoding. In sum, a code consists of the seed s , a one-time pad $\text{pad} = \mathcal{O}^{(t)}(s) \oplus (m||\rho)$, and a commitment $\text{com} = \mathcal{O}(s||m||\rho)$. Unfortunately, this construction is *still* malleable; it suffers the same mauling attack from Section 2.1. While it is hard for a depth $\ll t$ mauler M to compute $\mathcal{O}^{(t)}(s)$ for a random s , it can hardcode pairs $(s, \mathcal{O}^{(t)}(s))$ in its non-uniform advice, letting it generate a valid encoding $\tilde{c} = \text{Enc}(m)$ for any message m of its choice. Combined with the fact that the seed s and first bit of pad reveals part of the message to polynomial-time distinguishers, this gives a valid mauling attack.

Our final construction modifies the above approach very slightly but significantly. We add $\mathcal{O}^{(t)}(m)$ to the commitment, so $\text{com} = \mathcal{O}(s||m||\mathcal{O}^{(t)}(m)||\rho)$. This gives our code,

$$\text{Enc}^{\mathcal{O}}(m) := (s, \text{pad} = \mathcal{O}^{(t)}(s) \oplus (m||\rho), \text{com} = \mathcal{O}(s||m||\mathcal{O}^{(t)}(m)||\rho)).$$

To decode, you use the given seed s to unmask pad and get a candidate message and randomness $(\hat{m}||\rho) = \mathcal{O}^{(t)}(s) \oplus \text{pad}$. The decoding algorithm then outputs $m := \hat{m}$ if $\text{com} = \mathcal{O}(s||\hat{m}||\mathcal{O}^{(t)}(\hat{m})||\rho)$. The construction is rate-1 since s , ρ , and com can all have length independent of $|m|$.

We note that the slight modification above is inspired by the notion of a “hard-to-sample” function from [78] used in the rate-1 non-malleable code of [17]. At a very high level, it ensures that any mauler M can only compute $\text{Enc}(m)$ for a bounded set of messages m where the pair $(m, \mathcal{O}^{(t)}(m))$ is already hardcoded in its non-uniform advice. This limits the potential messages that can be “passed” to a distinguisher D , without relying on a long random string r as in the construction of Section 2.2.

Security analysis

Our high level goal is to argue that the output of $\mathcal{O}^{(t)}(s)$ can be lazily sampled independent to the rest of the non-malleability experiment for message m_b . If we can show this, we can switch to a hybrid experiment where pad is uniformly random and independent from m_b and ρ . But this implies that com also hides m_b , so any distinguisher D would have no advantage at guessing $\text{Enc}(m_b)$ for a random bit b . With this in mind, let’s take a closer look at the non-malleability experiment for $(\text{Enc}^{\mathcal{O}}, \text{Dec}^{\mathcal{O}})$.

- Non-Malleability Experiment for $(\text{Enc}^{\mathcal{O}}, \text{Dec}^{\mathcal{O}})$:
 1. A code $c = (s, \text{pad}, \text{com}) \leftarrow \text{Enc}^{\mathcal{O}}(m_b)$ is sampled for m_b .
 2. The mauler outputs a tampered codeword $\tilde{c} = (\tilde{s}, \tilde{\text{pad}}, \tilde{\text{com}}) \leftarrow M^{\mathcal{O}}(c)$.
 3. The MIM experiment outputs $\tilde{m} = \text{Dec}^{\mathcal{O}}(\tilde{c})$ as long as $\tilde{c} \neq c$.
 - Note that \tilde{m} is derived by unmasking $\tilde{\text{pad}}$ with $\mathcal{O}^{(t)}(\tilde{s})$.
 4. The distinguisher predicts a bit $b' = D^{\mathcal{O}}(\tilde{m})$ and wins if $b' = b$.

To argue that the experiment doesn’t query the output of $\mathcal{O}^{(t)}(s)$ computed by $\text{Enc}^{\mathcal{O}}(m_b)$, we separately argue that $M^{\mathcal{O}}$, $\text{Dec}^{\mathcal{O}}$, and $D^{\mathcal{O}}$ are unlikely to compute such a query.

- First, $M^{\mathcal{O}}$ runs in depth $\ll t$, so the only way it could possibly compute the output of $\mathcal{O}^{(t)}(s)$ is if it already had hardcoded advice about s . This is very unlikely, however, since s is randomly chosen in the experiment independent of $M^{\mathcal{O}}$.
- For $\text{Dec}^{\mathcal{O}}$, we consider two separate cases:
 - If $\tilde{s} \neq s$, we claim that the path of queries for $\mathcal{O}^{(t)}(\tilde{s})$ is very unlikely to collide with the path for $\mathcal{O}^{(t)}(s)$. For starters, we can domain separate the initial query to ensure that \tilde{s} is not already in the path of s . It follows that the values on the path queried by $\mathcal{O}^{(t)}(s)$ are all random and independent of $M^{\mathcal{O}}$, so $M^{\mathcal{O}}$ is very unlikely to have any hardcoded information on \mathcal{O} to come up with such a \tilde{s} .
 - If $\tilde{s} = s$, we observe that the experiment can derive the same value for \tilde{m} without computing $\mathcal{O}^{(t)}(s)$ at all! Formally, we switch to a hybrid where we use the facts that $\text{pad} = \mathcal{O}^{(t)}(s) \oplus (m_b || \rho)$ and $\tilde{\text{pad}} = \mathcal{O}^{(t)}(s) \oplus (\tilde{m} || \tilde{\rho})$ to instead derive \tilde{m} from $\tilde{\text{pad}} \oplus \text{pad} \oplus m_b$ since the $\mathcal{O}^{(t)}(s)$ terms cancel out. So the experiment doesn’t need to make any such queries to \mathcal{O} in this case.
- Finally, for $D^{\mathcal{O}}$, we leverage the fact that pairs $(m, \mathcal{O}^{(t)}(m))$ are “hard-to-sample” by any $M^{\mathcal{O}}$ running in depth $\ll t$. Intuitively, this means that $M^{\mathcal{O}}$ can only compute such pairs for a bounded, polynomial set of messages $M^{\mathcal{O}}$. Since $M^{\mathcal{O}}$ must compute $\text{com} = \mathcal{O}(\tilde{s} || \tilde{m} || \mathcal{O}^{(t)}(\tilde{m}) || \tilde{\rho})$ for any code \tilde{c} it outputs, it must already “know” the underlying message \tilde{m} (using the extractability of \mathcal{O}). It follows that $M^{\mathcal{O}}$ can only “pass” a polynomial-size set of messages to $D^{\mathcal{O}}$, who runs in polynomial time. This means there is only a polynomial set of values that $D^{\mathcal{O}}$ might have information on, so $D^{\mathcal{O}}$ is very unlikely to query the output of $\mathcal{O}^{(t)}(s)$ for a random s .

As such, the output of $\mathcal{O}^{(t)}(s)$ can be lazily sampled and independent of the rest of the experiment. We argued above that this suffices to guarantee that $D^{\mathcal{O}}$ has no advantage at predicting the challenge bit b , ruling out all possible mauling attacks.

On “hard-to-sample” functions

The new random oracle-based techniques in the construction and analysis above are inspired by the notion of a “hard-to-sample” function from [78] and used by [17]. In their work, they construct and use hard-to-sample functions in a complexity-theoretic setting that only achieves inverse-polynomial security. Here, we implicitly use such functions in our construction based on random oracles. We think it is a fascinating direction for future work to build cryptographically secure hard-to-sample functions in the plain model, and is one possible avenue towards constructing rate-1 non-malleable codes in the plain model with negligible security. However, we note that our proof above additionally relies on the extractability property of the random oracle when computing com .

3 Rate-1 Compiler for CCA-Security

In this section, we construct a rate-1 CCA-secure encoding scheme from one with inverse-polynomial rate, resulting in the following theorem.

► **Theorem 3.6** (Rate-1 Compiler for CCA-Secure Encoding Schemes). *Let $(\text{Enc}', \text{Dec}')$ be any encoding scheme that satisfies CCA-hiding for a subset of non-uniform PPT adversaries. For any $\text{aux} \in \{0, 1\}^*$, let $\beta(\lambda)$ be a bound on the size of $\text{Dec}'(c)$ for any $c \leftarrow \text{Enc}'(1^\lambda, \text{aux}, m)$ for $|m| \leq 3\lambda$. Assume the existence of β -secure collision-resistant hash functions. There exists a rate-1 encoding scheme (Enc, Dec) with a size-preserving straight-line, black-box reduction from the CCA-hiding of (Enc, Dec) to the CCA-hiding of $(\text{Enc}', \text{Dec}')$.*

Since the reduction in the above theorem is size-preserving, it preserves CCA-hiding for polynomial-size, bounded polynomial-depth, and bounded polynomial-size tampering classes.

We note that our construction relies on a CRH and a PRG, which are both known to be implied by CRH. At a high level, our construction uses the inner encoding Enc' to encode a PRG seed s and a collision-resistant hash of the message m . It then expands the seed s to mask the message m . We formalize this in Construction 3.7 below.

Construction 3.7 (Rate-1 CCA-Hiding Compiler).

Ingredients: Let G be pseudo-random generator, Hash be a collision-resistant hash function, and $(\text{Enc}', \text{Dec}')$ be any encoding scheme that satisfies CCA-hiding for some class of polynomially bounded adversaries.

Construction: We define encoding and decoding functions (Enc, Dec) as follows:

- $c \leftarrow \text{Enc}(1^\lambda, \text{aux}, m)$:
 1. Sample $s, \text{hk} \leftarrow \{0, 1\}^\lambda$.
 2. Compute $c' \leftarrow \text{Enc}'(1^\lambda, \text{aux}, (s, \text{hk}, \text{dig}))$ for $\text{dig} = \text{Hash}(\text{hk}, m)$.
 3. Set $\text{pad} = G(1^{|m|}, s) \oplus m$.
 4. Output $c := (c', \text{pad})$.
- $m = \text{Dec}(c)$:
 1. Parse $c := (c', \text{pad})$.
 2. Compute $(s, \text{hk}, \text{dig}) = \text{Dec}'(c')$.
 3. Set $\hat{m} = G(1^{|\text{pad}|}, s) \oplus \text{pad}$.
 4. Output $m := \hat{m}$ if $\text{Hash}(\text{hk}, \hat{m}) = \text{dig}$ and $m := \perp$ otherwise.

We prove that this construction satisfies the required properties for Theorem 3.6 in the full version.

4 Rate-1 Non-Malleable Code in the AI-ROM

In this section, we construct a rate-1 non-malleable code for bounded polynomial-depth adversaries in the AI-ROM, from no additional assumptions.

► **Theorem 4.8** (Non-Malleable Code for Bounded Polynomial-Depth Tampering in the AI-ROM). *In the AI-ROM, there exists a rate-1 non-malleable code that satisfied bounded polynomial-depth one-to-one non-malleability.*

Our construction for Theorem 4.8 makes use of a common random oracle \mathcal{O} . We'll leverage the fact that iterating a random oracle is an inherently sequential and “hard-to-sample” function. For any $t, n \in \mathbb{N}$, we define $\mathcal{O}^{(t)}(1^n, x)$ to be the t -fold composition of the function $\mathcal{O}(1^n, \cdot)$ starting with input x , so

$$\mathcal{O}^{(t)}(1^n, x) = \underbrace{\mathcal{O}(1^n, \cdot) \circ \dots \circ \mathcal{O}(1^n, \cdot)}_{t \text{ times}}(x).$$

At a high level, for a given time bound t , our encoding algorithm gives out a seed s in the clear and uses $\mathcal{O}^{(t)}(s)$ to mask the message m with randomness ρ . It then uses \mathcal{O} to produce a commitment for the relevant values s , ρ , and m as well as $\mathcal{O}^{(t)}(m)$. We formalize this in Construction 4.9 below.

Construction 4.9 (Non-Malleable Code in the AI-ROM).

Ingredients: Let \mathcal{O} be a common random function.

Construction: We construct encoding and decoding algorithms ($\text{Enc}^{\mathcal{O}}, \text{Dec}^{\mathcal{O}}$) as follows:

- $c \leftarrow \text{Enc}^{\mathcal{O}}(1^\lambda, t, m)$:
 1. Sample $s, \rho \leftarrow \{0, 1\}^\lambda$.
 2. Set $\text{pad} = y_s \oplus (m \parallel \rho)$ for $y_s = \mathcal{O}^{(t)}(1^{|m|+\lambda}, s)$.
 3. Compute $\text{com} = \mathcal{O}(1^\lambda, s \parallel \rho \parallel m \parallel y_m)$ for $y_m = \mathcal{O}^{(t)}(1^\lambda, m)$.
 4. Output $c = (s, \text{pad}, \text{com})$.
- $m = \text{Dec}^{\mathcal{O}}(c)$:
 1. Parse $c = (s, \text{pad}, \text{com})$ for parameters λ, t .
 2. Set $(\hat{m} \parallel \rho) = y_s \oplus \text{pad}$ for $y_s = \mathcal{O}^{(t)}(1^{|\text{pad}|}, s)$.
 3. Compute $y_{\hat{m}} = \mathcal{O}^{(t)}(1^\lambda, \hat{m})$.
 4. Output $m := \hat{m}$ if $\text{com} = \mathcal{O}(1^\lambda, s \parallel \rho \parallel \hat{m} \parallel y_{\hat{m}})$ and $m := \perp$ otherwise.

We prove that this construction satisfies the required properties for Theorem 4.8 in the full version.

5 Functional Non-Malleability from CCA-Hiding in the AI-ROM

In this section, we construct a functional non-malleable encoding scheme from a CCA-hiding one in the AI-ROM. We consider the following function class that allows for arbitrary functions with bounded output length.

► **Definition 5.10** (Function Class $\mathcal{F}_{k,\ell}$ with Bounded Output Length). Let $k: \mathbb{N} \rightarrow \mathbb{N}$ and $\ell: \mathbb{N}^3 \rightarrow \mathbb{N}$ be a function that may depend on k . Let $\mathcal{F}_{k,\ell}$ be a class of multi-input functions $f_{\lambda,t}$ parameterized by a security parameter $\lambda \in \mathbb{N}$ and auxiliary input $\mathbf{aux} \in \{0,1\}^*$. We say that $f_{\lambda,t} \in \mathcal{F}_{k,\ell}$ if for any $k' \leq k(\lambda)$, $n \in \mathbb{N}$, and messages $m_1, \dots, m_{k'} \in \{0,1\}^n$, it holds that $|f_{\lambda,t}(m_1, \dots, m_{k'})| \leq \ell(\lambda, |\mathbf{aux}|, n)$.

We prove the following theorem in the rest of this section.

► **Theorem 5.11** (Functional Non-Malleability Compiler for CCA-Hiding Encoding Schemes). Let $(\text{Enc}', \text{Dec}')$ be any encoding scheme that satisfies CCA-hiding for a subset of non-uniform PPT adversaries. For any efficiently computable $k: \mathbb{N} \rightarrow \mathbb{N}$ and $\ell: \mathbb{N}^3 \rightarrow \mathbb{N}$, there exists an encoding scheme (Enc, Dec) in the AI-ROM with rate $\alpha(\lambda, |\mathbf{aux}|, n) = 1/(1 + \ell(\lambda, |\mathbf{aux}|, n)/n)$ with a depth-preserving reduction from the functional non-malleability of (Enc, Dec) for $\mathcal{F}_{k,\ell}$ to the CCA-hiding of $(\text{Enc}', \text{Dec}')$.

Before giving the construction, we remark on a few immediate corollaries to Theorem 5.11 that follow from instantiating the theorem with particular function classes.

- For plain (one-to-one) non-malleability, $k(\lambda) = 1$ and $\ell(\lambda, |\mathbf{aux}|, n) = n$, so the resulting construction has rate $1/2$.
- For k -bounded concurrent non-malleability, $\ell(\lambda, |\mathbf{aux}|, n) = k(\lambda) \cdot n$, so the resulting construction has rate $1/(1 + k)$. For the case of time-lock puzzles, [40] prove that any k -bounded concurrent non-malleable time-lock puzzle has rate at most $1/k$, so our compiler nearly matches this bound.
- Finally, the application to multi-party coin-flipping requires functional non-malleability for the “ \oplus ” function, which has $\ell(\lambda, |\mathbf{aux}|, n) = n$ for any (potentially unbounded) k . For this function class, our resulting construction has rate $1/2$. In comparison, the corresponding functional non-malleable time-lock puzzle of [40] in the AI-ROM only achieves inverse polynomial-rate.

► **Remark 5.12** (On allowing functions that query \mathcal{O}). We note that the Theorem 5.11 holds for all (potentially unbounded) functions $f \in \mathcal{F}_{k,\ell}$ as long as the output length is bounded. However, we do not allow functions f that query \mathcal{O} . This is still meaningful as the function class $\mathcal{F}_{k,\ell}$ itself is not adversarially chosen and still captures many of the above relevant application scenarios. With a slight modification to the proof below, we could allow f to query \mathcal{O} , but we would then require f to be restricted in the same way as the MIM adversary in the functional non-malleability game.

Our construction for Theorem 5.11 makes use of an underlying encoding scheme $(\text{Enc}', \text{Dec}')$ and a common random oracle \mathcal{O} . At a high level, our construction uses the inner encoding Enc' to encode a small seed s and a succinct commitment to m using randomness ρ (based on \mathcal{O}). It additionally outputs a sufficiently long random string r . It then uses \mathcal{O} to expand the seed s together with r to mask the message m and ρ . We formalize this in Construction 5.13 below.

Construction 5.13 (Functional NM Codes from CCA-Hiding).

Ingredients: Let $(\text{Enc}', \text{Dec}')$ be any encoding scheme that satisfies CCA-hiding for some class of polynomially bounded adversaries, and let \mathcal{O} be a common random function.

Construction: For each efficiently computable k, ℓ , we construct encoding and decoding algorithms $(\text{Enc}_{k,\ell}^{\mathcal{O}}, \text{Dec}_{k,\ell}^{\mathcal{O}})$ as follows:

- $c \leftarrow \text{Enc}_{k,\ell}^{\mathcal{O}}(1^\lambda, \text{aux}, m)$:
 1. Sample $s, \text{hk}, \rho \leftarrow \{0, 1\}^\lambda$ and $r \leftarrow \{0, 1\}^{\ell(\lambda, |\text{aux}|, |m|)}$.
 2. Compute $c' \leftarrow \text{Enc}'(1^\lambda, \text{aux}, (s, \text{hk}, \text{com}))$ for $\text{com} = \mathcal{O}(1^\lambda, \text{hk}||m||\rho)$.
 3. Set $\text{pad} = \mathcal{O}(1^{|m|+\lambda}, s||r) \oplus (m||\rho)$.
 4. Output $c := (c', r, \text{pad})$.
- $m = \text{Dec}_{k,\ell}^{\mathcal{O}}(c)$:
 1. Parse $c := (c', r, \text{pad})$.
 2. Compute $(s, \text{hk}, \text{com}) = \text{Dec}'(c')$.
 3. Set $(\hat{m}||\rho) = \mathcal{O}(1^{|\text{pad}|}, s||r) \oplus \text{pad}$.
 4. Output $m := \hat{m}$ if $\text{com} = \mathcal{O}(1^\lambda, \text{hk}||\hat{m}||\rho)$ and $m := \perp$ otherwise.

We prove that this construction satisfies the required properties for Theorem 5.11 in the full version.

5.1 Random Oracle Uninstantiability for Fully Homomorphic TLPs

In this section, we exhibit a random oracle uninstantiability result for our transformation of Theorem 5.11. Specifically, we show sufficient conditions for the underlying CCA-hiding time-lock puzzle such that the resulting transformation is malleable for any fixed polynomial-size instantiation H of the random oracle \mathcal{O} .

We first define the additional properties needed for the underlying time-lock puzzle. At a high level, we require that the underlying time-lock puzzle is fully homomorphic, similar to the notion introduced by [69]. Specifically, we require that there is an associated evaluation algorithm Eval that mauls an encodings c for a message m into new (invalid) encodings c^* corresponding to some function $f(m)$ that can be recovered in time roughly t . Since c^* is allowed to be an invalid encoding, this notion is not in conflict with the notion of CCA hiding. We model this via an alternative decoding algorithm Dec_{alt} for evaluated encodings.

► **Definition 5.14** (Fully Homomorphic TLPs). *Let \mathcal{C} be a class of circuits. We say that a time-lock puzzle (Enc, Dec) is homomorphic for \mathcal{C} if there exist algorithms $(\text{Eval}, \text{Dec}_{\text{alt}})$ with the following syntax:*

- $c^* = \text{Eval}(f, c)$: *A deterministic algorithm that on input a circuit $f \in \mathcal{C}$ and encoding $c \in \{0, 1\}^*$ outputs a new encoding c^* .*
- $m^* = \text{Dec}_{\text{alt}}(c^*)$: *A deterministic algorithm that on input an encoding $c^* \in \{0, 1\}^*$ outputs a message m^* .*

We require that $(\text{Eval}, \text{Dec}_{\text{alt}})$ additionally satisfy the following properties:

- **Evaluation Efficiency**: *there exists a polynomial q such that $\text{Eval}(f, c)$ runs in time $q(|f|, |c|)$ and $\text{Dec}_{\text{alt}}(c^*)$ runs in time $t \cdot \text{poly}(|c|)$ for encodings c^* that specify the time bound t .*
- **Correctness of Evaluation**: *For any $\lambda, t \in \mathbb{N}$, $m \in \{0, 1\}^*$, and $f \in \mathcal{C}$, it holds that*

$$\Pr \left[\begin{array}{l} c \leftarrow \text{Enc}(1^\lambda, t, m) \\ c^* = \text{Eval}(f, c) \\ m^* = \text{Dec}_{\text{alt}}(c^*) \end{array} : m^* = f(m) \right] = 1.$$

If (Enc, Dec) is homomorphic for the class of all polynomial-size circuits, we say that it is fully homomorphic.

We give our formal random oracle uninstantiability result in the following theorem.

► **Theorem 5.15.** *Let $(\text{Enc}', \text{Dec}')$ be a fully homomorphic TLP with associated algorithms $(\text{Eval}, \text{Dec}_{\text{alt}})$. Let H be any polynomial-size hash function. Then the corresponding TLP construction $(\text{Enc}_{1,|m|}^H, \text{Dec}_{1,|m|}^H)$ from Construction 13 does not satisfy non-malleability.*

Proof. We construct a mauling attacker $(Z_\lambda, M_\lambda, D_\lambda)$ for each $\lambda \in \mathbb{N}$ and any sufficiently large message length $n_\lambda \in \mathbb{N}$ specified below:

- Z_λ outputs $(m_0, m_1, t_\lambda, f_{\text{id}})$ where $m_0 = 0^{n_\lambda}$, $m_1 = 1^{n_\lambda}$, $t_\lambda \in \mathbb{N}$ is larger than the description of M_λ below, and f_{id} is the identity function of n_λ -bit strings.
- M_λ takes as input an encoding $c = (c', r, \text{pad})$ and does the following:
 - Compute $c^* = \text{Eval}(f_{r, \text{pad}}, c')$ for function $f_{r, \text{pad}}(s, \text{hk}, \text{com})$ that outputs the first bit of $H(1^{n_\lambda + \lambda}, s || r) \oplus \text{pad}$.
 - Output $\tilde{c} \leftarrow \text{Enc}(c^* || 0^{n_\lambda - |c^*|})$. We require that $n_\lambda \geq |c^*|$.
- D_λ receives as input a string $\tilde{m} = c^* || 0^{n_\lambda - |c^*|}$ and outputs $\text{Dec}_{\text{alt}}(c^*)$.

It remains to argue that $(Z_\lambda, M_\lambda, D_\lambda)$ is a valid mauling attacker.

First, in terms of efficiency, we note that $n_\lambda \in \text{poly}(\lambda)$ and independent of t_λ by the efficiency of Eval as $|f_{r, \text{pad}}|$ and $|c'|$ are independent of t_λ . Thus, it also holds that $t_\lambda \in \text{poly}(\lambda)$ by the efficiency of Eval and Enc , and furthermore $|M_\lambda| < t_\lambda$ by construction. Lastly, D_λ runs in time $t_\lambda \cdot \text{poly}(\lambda)$ by the efficiency of Dec_{alt} . It follows that $(Z_\lambda, M_\lambda, D_\lambda)$ has the required efficiency.

In terms of correctness, suppose that $c \leftarrow \text{Enc}(1^\lambda, t, m_b)$ for some bit $b \in \{0, 1\}$. We observe that the value $m^* = \text{Dec}_{\text{alt}}(c^*)$ output by D_λ is equal to $f_{r, \text{pad}}(s, \text{hk}, \text{com})$ by construction. Note that $f_{r, \text{pad}}$ simply unmaskes the first bit of m_b , which is simply equal to b by construction. It follows that D_λ always outputs $b' = b$ in the non-malleability experiment, so it is a valid attack. ◀

Discussion of uninstantiability result

We note that Theorem 5.15 gives sufficient conditions for which the transformation of Theorem 5.11 is malleable for any concrete instantiation of the random oracle. As a result, one could assume that the transformation is secure for “natural” TLP candidates that don’t clearly support homomorphic properties. Indeed, existing candidates of fully homomorphic time-lock puzzles seem to rely on strong tools like fully homomorphic encryption or obfuscation [69] and are only proven secure with setup in the CRS model, so it is potentially a reasonable assumption that the compiler is secure for non-contrived CCA-hiding TLPs. This is similar to real-world applications of the Fiat-Shamir heuristic for arguments [39] or the Fujisaki-Okamoto transform for CCA-secure encryption [41], where we know there are concrete counterexamples [18, 44, 20, 57, 25, 45], yet it is believed to still be secure for many practical applications.

Concretely, we observe that the proof of Theorem 5.11 avoids this attack by the extractability property of a random oracle \mathcal{O} . Namely, the only way to compute $\mathcal{O}(s || r)$ is by “knowing” s . However, the attack from Theorem 5.15 shows this is not true if you can homomorphically evaluate $H(s || r)$ *without knowing* s for any polynomial-size instantiation H for \mathcal{O} .

To the best of our knowledge, a similar attack does not apply to our non-malleable code of Section 4 as it is proven unconditionally secure in the AI-ROM without any additional assumptions. Furthermore, the implicit underlying code that computes $y_s = \mathcal{O}^{(t)}(s)$ lacks structure and is very unlikely to satisfy any homomorphic properties.

Overall, we view the compiler in this section as a first step towards a more efficient non-malleable TLP construction in the plain model, highlighting many of the technical and definitional challenges involved in using and composing time-lock puzzles.

References

- 1 Blockchain data storage. <https://ethereum.org/en/developers/docs/data-availability/blockchain-data-storage-strategies/#storage>. Accessed: 2025-02-11.
- 2 Divesh Aggarwal, Shashank Agrawal, Divya Gupta, Hemanta K. Maji, Omkant Pandey, and Manoj Prabhakaran. Optimal computational split-state non-malleable codes. In *Theory of Cryptography - 13th International Conference, TCC*, pages 393–417, 2016. doi:10.1007/978-3-662-49099-0_15.
- 3 Divesh Aggarwal, Yevgeniy Dodis, Tomasz Kazana, and Maciej Obremski. Non-malleable reductions and applications. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC*, pages 459–468, 2015. doi:10.1145/2746539.2746544.
- 4 Divesh Aggarwal, Yevgeniy Dodis, and Shachar Lovett. Non-malleable codes from additive combinatorics. *SIAM J. Comput.*, 47(2):524–546, 2018. doi:10.1137/140985251.
- 5 Divesh Aggarwal, Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, Maciej Obremski, and Sruthi Sekar. Rate one-third non-malleable codes. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 1364–1377. ACM, 2022. doi:10.1145/3519935.3519972.
- 6 Shashank Agrawal, Divya Gupta, Hemanta K. Maji, Omkant Pandey, and Manoj Prabhakaran. A rate-optimizing compiler for non-malleable codes against bit-wise tampering and permutations. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part I*, volume 9014 of *Lecture Notes in Computer Science*, pages 375–397. Springer, 2015. doi:10.1007/978-3-662-46494-6_16.
- 7 Knud Ahrens. SIGNITC: supersingular isogeny graph non-interactive timed commitments. *IACR Cryptol. ePrint Arch.*, page 1225, 2024. URL: <https://eprint.iacr.org/2024/1225>.
- 8 Andreea B. Alexandru, Julian Loss, Charalampos Papamanthou, Giorgos Tsimos, and Benedikt Wagner. Sublinear-round broadcast without trusted setup. *IACR Cryptol. ePrint Arch.*, page 770, 2024. URL: <https://eprint.iacr.org/2024/770>.
- 9 Marshall Ball, Eshan Chattopadhyay, Jun-Jie Liao, Tal Malkin, and Li-Yang Tan. Non-malleability against polynomial tampering. In *CRYPTO (3)*, volume 12172 of *Lecture Notes in Computer Science*, pages 97–126. Springer, 2020. doi:10.1007/978-3-030-56877-1_4.
- 10 Marshall Ball, Dana Dachman-Soled, Siyao Guo, Tal Malkin, and Li-Yang Tan. Non-malleable codes for small-depth circuits. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 826–837, 2018. doi:10.1109/FOCS.2018.00083.
- 11 Marshall Ball, Dana Dachman-Soled, Mukul Kulkarni, Huijia Lin, and Tal Malkin. Non-malleable codes against bounded polynomial time tampering. In *Advances in Cryptology - EUROCRYPT*, pages 501–530, 2019. doi:10.1007/978-3-030-17653-2_17.
- 12 Marshall Ball, Dana Dachman-Soled, Mukul Kulkarni, and Tal Malkin. Non-malleable codes for bounded depth, bounded fan-in circuits. In *Advances in Cryptology - EUROCRYPT*, pages 881–908, 2016. doi:10.1007/978-3-662-49896-5_31.
- 13 Marshall Ball, Dana Dachman-Soled, Mukul Kulkarni, and Tal Malkin. Non-malleable codes for bounded depth, bounded fan-in circuits. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 881–908. Springer, 2016. doi:10.1007/978-3-662-49896-5_31.
- 14 Marshall Ball, Dana Dachman-Soled, Mukul Kulkarni, and Tal Malkin. Non-malleable codes from average-case hardness: AC^0 , decision trees, and streaming space-bounded tampering. In *Advances in Cryptology - EUROCRYPT*, pages 618–650, 2018.
- 15 Marshall Ball, Dana Dachman-Soled, and Julian Loss. (nondeterministic) hardness vs. non-malleability. In *CRYPTO (1)*, volume 13507 of *Lecture Notes in Computer Science*, pages 148–177. Springer, 2022. doi:10.1007/978-3-031-15802-5_6.

- 16 Marshall Ball, Siyao Guo, and Daniel Wichs. Non-malleable codes for decision trees. In *Advances in Cryptology - CRYPTO*, pages 413–434, 2019. doi:10.1007/978-3-030-26948-7_15.
- 17 Marshall Ball, Ronen Shaltiel, and Jad Silbak. Non-malleable codes with optimal rate for poly-size circuits. In *EUROCRYPT (4)*, volume 14654 of *Lecture Notes in Computer Science*, pages 33–54. Springer, 2024. doi:10.1007/978-3-031-58737-5_2.
- 18 Boaz Barak. How to go beyond the black-box simulation barrier. In *42nd Annual Symposium on Foundations of Computer Science, FOCS*, pages 106–115, 2001. doi:10.1109/SFCS.2001.959885.
- 19 Boaz Barak. Constant-round coin-tossing with a man in the middle or realizing the shared random string model. In *43rd Symposium on Foundations of Computer Science FOCS*, pages 345–355, 2002. doi:10.1109/SFCS.2002.1181957.
- 20 James Bartusek, Liron Bronfman, Justin Holmgren, Fermi Ma, and Ron D. Rothblum. On the (in)security of kilian-based snargs. In *TCC (2)*, volume 11892 of *Lecture Notes in Computer Science*, pages 522–551. Springer, 2019. doi:10.1007/978-3-030-36033-7_20.
- 21 Carsten Baum, Bernardo David, Rafael Dowsley, Jesper Buus Nielsen, and Sabine Oechsner. TARDIS: A foundation of time-lock puzzles in UC. In *EUROCRYPT (3)*, volume 12698 of *Lecture Notes in Computer Science*, pages 429–459. Springer, 2021. doi:10.1007/978-3-030-77883-5_15.
- 22 Nir Bitansky and Huijia Lin. One-message zero knowledge and non-malleable commitments. In *Theory of Cryptography - 16th International Conference, TCC*, pages 209–234, 2018. doi:10.1007/978-3-030-03807-6_8.
- 23 Dan Boneh and Moni Naor. Timed commitments. In *CRYPTO*, volume 1880 of *Lecture Notes in Computer Science*, pages 236–254. Springer, 2000. doi:10.1007/3-540-44598-6_15.
- 24 Zvika Brakerski, Shai Halevi, and Antigoni Polychroniadou. Four round secure computation without setup. In *TCC (1)*, volume 10677 of *Lecture Notes in Computer Science*, pages 645–677. Springer, 2017. doi:10.1007/978-3-319-70500-2_22.
- 25 Christina Brzuska, Pooya Farshim, and Arno Mittelbach. Random-oracle uninstantiability from indistinguishability obfuscation. In *TCC (2)*, volume 9015 of *Lecture Notes in Computer Science*, pages 428–455. Springer, 2015. doi:10.1007/978-3-662-46497-7_17.
- 26 Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In *STOC*, pages 209–218. ACM, 1998. doi:10.1145/276698.276741.
- 27 Ran Canetti, Huijia Lin, and Rafael Pass. Adaptive hardness and composable security in the plain model from standard assumptions. In *FOCS*, pages 541–550. IEEE Computer Society, 2010. doi:10.1109/FOCS.2010.86.
- 28 Eshan Chattopadhyay and Xin Li. Non-malleable codes and extractors for small-depth circuits, and affine functions. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 1171–1184, 2017. doi:10.1145/3055399.3055483.
- 29 Peter Chvojka and Tibor Jager. Simple, fast, efficient, and tightly-secure non-malleable non-interactive timed commitments. In *Public Key Cryptography (1)*, volume 13940 of *Lecture Notes in Computer Science*, pages 500–529. Springer, 2023. doi:10.1007/978-3-031-31368-4_18.
- 30 Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Ivan Visconti. Concurrent non-malleable commitments (and more) in 3 rounds. In *Advances in Cryptology - CRYPTO*, pages 270–299, 2016. doi:10.1007/978-3-662-53015-3_10.
- 31 Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Ivan Visconti. Four-round concurrent non-malleable commitments from one-way functions. In *Advances in Cryptology - CRYPTO*, pages 127–157, 2017. doi:10.1007/978-3-319-63715-0_5.
- 32 Sandro Coretti, Antonio Faonio, and Daniele Venturi. Rate-optimizing compilers for continuously non-malleable codes. In *ACNS*, volume 11464 of *Lecture Notes in Computer Science*, pages 3–23. Springer, 2019. doi:10.1007/978-3-030-21568-2_1.

- 33 Dana Dachman-Soled, Ilan Komargodski, and Rafael Pass. Non-malleable codes for bounded parallel-time tampering. In *Advances in Cryptology - CRYPTO*, pages 535–565, 2021. doi:10.1007/978-3-030-84252-9_18.
- 34 Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, STOC*, pages 542–552, 1991. doi:10.1145/103418.103474.
- 35 Stefan Dziembowski, Krzysztof Pietrzak, and Daniel Wichs. Non-malleable codes. In *Innovations in Computer Science - ICS*, pages 434–452, 2010. URL: <http://conference.iis.tsinghua.edu.cn/ICS2010/content/papers/34.html>.
- 36 Karim Eldefrawy, Benjamin Terner, and Moti Yung. Composing timed cryptographic protocols: Foundations and applications. *IACR Cryptol. ePrint Arch.*, page 676, 2024. URL: <https://eprint.iacr.org/2024/676>.
- 37 Sebastian Faust, Kristina Hostáková, Pratyay Mukherjee, and Daniele Venturi. Non-malleable codes for space-bounded tampering. In *Advances in Cryptology - CRYPTO*, pages 95–126, 2017. doi:10.1007/978-3-319-63715-0_4.
- 38 Sebastian Faust, Pratyay Mukherjee, Daniele Venturi, and Daniel Wichs. Efficient non-malleable codes and key derivation for poly-size tampering circuits. *IEEE Trans. Information Theory*, 62(12):7179–7194, 2016. doi:10.1109/TIT.2016.2613919.
- 39 Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986. doi:10.1007/3-540-47721-7_12.
- 40 Cody Freitag, Ilan Komargodski, Rafael Pass, and Naomi Sirkin. Non-malleable time-lock puzzles and applications. In *Theory of Cryptography - 19th International Conference, TCC*, pages 447–479, 2021. doi:10.1007/978-3-030-90456-2_15.
- 41 Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Advances in Cryptology - CRYPTO*, pages 537–554, 1999. doi:10.1007/3-540-48405-1_34.
- 42 Rachit Garg, Dakshita Khurana, George Lu, and Brent Waters. Black-box non-interactive non-malleable commitments. In *EUROCRYPT (3)*, volume 12698 of *Lecture Notes in Computer Science*, pages 159–185. Springer, 2021. doi:10.1007/978-3-030-77883-5_6.
- 43 Rachit Garg, Dakshita Khurana, George Lu, and Brent Waters. On non-uniform security for black-box non-interactive CCA commitments. In *EUROCRYPT (1)*, volume 14004 of *Lecture Notes in Computer Science*, pages 173–204. Springer, 2023. doi:10.1007/978-3-031-30545-0_7.
- 44 Shafi Goldwasser and Yael Tauman Kalai. On the (in)security of the fiat-shamir paradigm. In *44th Symposium on Foundations of Computer Science (FOCS 2003)*, pages 102–113, 2003. doi:10.1109/SFCS.2003.1238185.
- 45 Rishab Goyal, Venkata Koppula, and Brent Waters. Lockable obfuscation. In *FOCS*, pages 612–621. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.62.
- 46 Vipul Goyal. Constant round non-malleable protocols using one way functions. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC*, pages 695–704, 2011. doi:10.1145/1993636.1993729.
- 47 Vipul Goyal, Chen-Kuei Lee, Rafail Ostrovsky, and Ivan Visconti. Constructing non-malleable commitments: A black-box approach. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS*, pages 51–60, 2012. doi:10.1109/FOCS.2012.47.
- 48 Vipul Goyal, Huijia Lin, Omkant Pandey, Rafael Pass, and Amit Sahai. Round-efficient concurrently composable secure computation via a robust extraction lemma. In *TCC (1)*, volume 9014 of *Lecture Notes in Computer Science*, pages 260–289. Springer, 2015. doi:10.1007/978-3-662-46494-6_12.
- 49 Vipul Goyal, Omkant Pandey, and Silas Richelson. Textbook non-malleable commitments. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 1128–1141, 2016. doi:10.1145/2897518.2897657.

- 50 Vipul Goyal and Silas Richelson. Non-malleable commitments using goldreich-levin list decoding. In *FOCS*, pages 686–699. IEEE Computer Society, 2019. doi:10.1109/FOCS.2019.00047.
- 51 Divya Gupta, Hemanta K. Maji, and Mingyuan Wang. Constant-rate non-malleable codes in the split-state model. *IACR Cryptol. ePrint Arch.*, page 1048, 2017. URL: <http://eprint.iacr.org/2017/1048>.
- 52 Divya Gupta, Hemanta K. Maji, and Mingyuan Wang. Explicit rate-1 non-malleable codes for local tampering. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part I*, volume 11692 of *Lecture Notes in Computer Science*, pages 435–466. Springer, 2019. doi:10.1007/978-3-030-26948-7_16.
- 53 Yael Tauman Kalai and Dakshita Khurana. Non-interactive non-malleability from quantum supremacy. In *Advances in Cryptology - CRYPTO*, pages 552–582, 2019. doi:10.1007/978-3-030-26954-8_18.
- 54 Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, and Sruthi Sekar. Four-state non-malleable codes with explicit constant rate. In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part II*, volume 10678 of *Lecture Notes in Computer Science*, pages 344–375. Springer, 2017. doi:10.1007/978-3-319-70503-3_11.
- 55 Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, and Sruthi Sekar. Non-malleable randomness encoders and their applications. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part III*, volume 10822 of *Lecture Notes in Computer Science*, pages 589–617. Springer, 2018. doi:10.1007/978-3-319-78372-7_19.
- 56 Jonathan Katz, Julian Loss, and Jiayu Xu. On the security of time-lock puzzles and timed commitments. In *TCC (3)*, volume 12552 of *Lecture Notes in Computer Science*, pages 390–413. Springer, 2020. doi:10.1007/978-3-030-64381-2_14.
- 57 Dmitry Khovratovich, Ron D. Rothblum, and Lev Soukhanov. How to prove false statements: Practical attacks on fiat-shamir. In *CRYPTO (6)*, volume 16005 of *Lecture Notes in Computer Science*, pages 3–26. Springer, 2025. doi:10.1007/978-3-032-01887-8_1.
- 58 Dakshita Khurana. Round optimal concurrent non-malleability from polynomial hardness. In *Theory of Cryptography - 15th International Conference, TCC*, pages 139–171, 2017. doi:10.1007/978-3-319-70503-3_5.
- 59 Dakshita Khurana. Non-interactive distributional indistinguishability (NIDI) and non-malleable commitments. In *EUROCRYPT (3)*, volume 12698 of *Lecture Notes in Computer Science*, pages 186–215. Springer, 2021. doi:10.1007/978-3-030-77883-5_7.
- 60 Dakshita Khurana and Amit Sahai. How to achieve non-malleability in one or two rounds. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 564–575, 2017. doi:10.1109/FOCS.2017.58.
- 61 Susumu Kiyoshima. Round-efficient black-box construction of composable multi-party computation. In *CRYPTO (2)*, volume 8617 of *Lecture Notes in Computer Science*, pages 351–368. Springer, 2014. doi:10.1007/978-3-662-44381-1_20.
- 62 Huijia Lin and Rafael Pass. Non-malleability amplification. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC*, pages 189–198, 2009. doi:10.1145/1536414.1536442.
- 63 Huijia Lin and Rafael Pass. Constant-round non-malleable commitments from any one-way function. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC*, pages 705–714, 2011. doi:10.1145/1993636.1993730.
- 64 Huijia Lin and Rafael Pass. Black-box constructions of composable protocols without set-up. In *CRYPTO*, volume 7417 of *Lecture Notes in Computer Science*, pages 461–478. Springer, 2012. doi:10.1007/978-3-642-32009-5_27.

- 65 Huijia Lin, Rafael Pass, and Pratik Soni. Two-round and non-interactive concurrent non-malleable commitments from time-lock puzzles. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 576–587. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.59.
- 66 Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkatasubramaniam. Concurrent non-malleable commitments from any one-way function. In *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC*, pages 571–588, 2008. doi:10.1007/978-3-540-78524-8_31.
- 67 Feng-Hao Liu and Anna Lysyanskaya. Tamper and leakage resilience in the split-state model. In *Advances in Cryptology - CRYPTO*, pages 517–532, 2012. doi:10.1007/978-3-642-32009-5_30.
- 68 Mohammad Mahmoody, Tal Moran, and Salil P. Vadhan. Time-lock puzzles in the random oracle model. In *CRYPTO*, volume 6841 of *Lecture Notes in Computer Science*, pages 39–50. Springer, 2011. doi:10.1007/978-3-642-22792-9_3.
- 69 Giulio Malavolta and Sri Aravinda Krishnan Thyagarajan. Homomorphic time-lock puzzles and applications. In *Advances in Cryptology - CRYPTO*, pages 620–649, 2019. doi:10.1007/978-3-030-26948-7_22.
- 70 Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *STOC*, pages 427–437. ACM, 1990. doi:10.1145/100216.100273.
- 71 Omkant Pandey, Rafael Pass, and Vinod Vaikuntanathan. Adaptive one-way functions and applications. In *Advances in Cryptology - CRYPTO*, pages 57–74, 2008. doi:10.1007/978-3-540-85174-5_4.
- 72 Rafael Pass. Unprovable security of perfect NIZK and non-interactive non-malleable commitments. In *TCC*, volume 7785 of *Lecture Notes in Computer Science*, pages 334–354. Springer, 2013. doi:10.1007/978-3-642-36594-2_19.
- 73 Rafael Pass and Alon Rosen. Concurrent non-malleable commitments. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 563–572, 2005. doi:10.1109/SFCS.2005.27.
- 74 Rafael Pass and Alon Rosen. New and improved constructions of non-malleable cryptographic protocols. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, STOC*, pages 533–542, 2005. doi:10.1145/1060590.1060670.
- 75 Rafael Pass and Hoeteck Wee. Constant-round non-malleable commitments from sub-exponential one-way functions. In *Advances in Cryptology - EUROCRYPT*, 2010.
- 76 Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *CRYPTO*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444. Springer, 1991. doi:10.1007/3-540-46766-1_35.
- 77 Ronald L Rivest, Adi Shamir, and David A Wagner. Time-lock puzzles and timed-release crypto, 1996.
- 78 Ronen Shaltiel and Jad Silbak. Explicit codes for poly-size circuits and functions that are hard to sample on low entropy distributions. In *STOC*, pages 2028–2038. ACM, 2024. doi:10.1145/3618260.3649735.
- 79 Sri Aravinda Krishnan Thyagarajan, Guilhem Castagnos, Fabien Laguillaumie, and Giulio Malavolta. Efficient CCA timed commitments in class groups. In *CCS*, pages 2663–2684. ACM, 2021. doi:10.1145/3460120.3484773.
- 80 Nirvan Tyagi, Arasu Arun, Cody Freitag, Riad Wahby, Joseph Bonneau, and David Mazières. Riggs: Decentralized sealed-bid auctions. In *CCS*, pages 1227–1241. ACM, 2023. doi:10.1145/3576915.3623182.
- 81 Dominique Unruh. Random oracles and auxiliary input. In *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings*, pages 205–223, 2007. doi:10.1007/978-3-540-74143-5_12.
- 82 Hoeteck Wee. Black-box, round-efficient secure computation via non-malleability amplification. In *FOCS*, 2010.