Message Franking via Committing Authenticated Encryption

Paul Grubbs[1], Jiahui Lu[2], and Thomas Ristenpart[1]

[1] Cornell Tech
[2] Shanghai Jiao Tong University

Abstract. We initiate the study of message franking, recently introduced in Facebook's end-to-end encrypted message system. It targets verifiable reporting of abusive messages to Facebook without compromising security guarantees. We capture the goals of message franking via a new cryptographic primitive: compactly committing authenticated encryption with associated data (AEAD). This is an AEAD scheme for which a small part of the ciphertext can be used as a cryptographic commitment to the message contents. Decryption provides, in addition to the message, a value that can be used to open the commitment. Security for franking mandates more than that required of traditional notions associated with commitment. Nevertheless, and despite the fact that AEAD schemes are in general not committing (compactly or otherwise), we prove that many in-use AEAD schemes can be used for message franking by using secret keys as openings. An implication of our results is the first proofs that several in-use symmetric encryption schemes are committing in the traditional sense. We also propose and analyze schemes that retain security even after openings are revealed to an adversary. One is a generalization of the scheme implicitly underlying Facebook's message franking protocol, and another is a new construction that offers improved performance.

Keywords: authenticated encryption, encrypted messaging

1 Introduction

Encrypted messaging systems are now used by more than a billion people, due to the introduction of popular, industry-promoted products including WhatsApp [60], Signal [61], and Facebook Messenger [30]. These use specialized (non-interactive) key exchange protocols, in conjunction with authenticated encryption, to protect messages. Many tools are based off the Signal protocol [44], which itself was inspired by elements of the off-the-record (OTR) messaging protocol [20]. A primary design goal is end-to-end security: intermediaries including the messaging service providers, or those with access to their systems, should not be able to violate confidentiality or integrity of user messages.

End-to-end security can be at odds with other security goals. A well known example is dealing with filtering and reporting spam in the context of encrypted email [39, 56]. Similar issues arise in modern encrypted messaging systems. For

Scheme	MO security	Sender binding	Rec. binding	Enc	Dec	Ver
Encode-then-Encipher (Ideal)		✓	✓	–	–	–
Encrypt-then-HMAC (one key)		✓	✓	2+1	2+1	2+1
HMAC-then-CBC		✓	✓	2+1	2+1	2+1
CtE1	✓	✓	✓	3+1	3+1	1+1
CtE2 (Facebook)	✓	✓	✓	3+2	3+2	1+1
CEP	✓	✓	✓	2+1	2+1	1+1

Fig. 1. Summary of schemes investigated in this work. The columns indicate whether the scheme meets multiple-opening (MO) security, sender binding, and receiver binding. The last three columns indicate the number of cryptographic passes over a bit string of length equal to the message plus the number of passes needed to handle the associated data, for each of the three main operations. We omit comparisons with concrete encode-then-encipher constructions, which vary in the number of passes required.

example, in Facebook's system when one user sends harassing messages, phishing links, malware attachments, etc., the recipient should be able to report the malicious behavior so that Facebook can block or otherwise penalize the sender. But end-to-end confidentiality means that Facebook must rely on users sending examples of malicious messages. How can the provider know that the reported message was the one sent? Reports could, in turn, become a vector for abuse should they allow a malicious reporter to fabricate a message and convince the provider it was the one sent [39].

Facebook messenger recently introduced a seeming solution for verifiable abuse reporting that they refer to as *message franking* [31, 47]. The idea is to include in the report a cryptographic proof that the reported message was the one sent, encrypted, by the particular sender. They offer a protocol (discussed below) and a sensible, but informal and vague, discussion of security goals. At present it is ultimately not clear what message franking provides, whether their approach is secure, and if there exist better constructions. Given the critical role message franking will play for most messaging services moving forward, more study is clearly needed.

We therefore initiate the formal study of message franking. We introduce the notion of compactly committing authenticated encryption with associated data (AEAD) as the cryptographic primitive of merit that serves as the basis for message franking. We provide security definitions, show how several widely used existing AEAD schemes can already serve as compactly committing AEAD, give an analysis of (a generalization of) the scheme underlying Facebook's protocol, and design a new scheme that has superior performance. A summary of schemes treated in this work, and their efficiency, is shown in Figure 1.

Facebook's message franking protocol. Facebook's protocol works as follows, modulo a few details (see Section 3). A sender first generates a fresh key for HMAC [3], and applies HMAC to the message. It then encrypts the HMAC key and message using a conventional AEAD scheme with a symmetric key shared with (just) the recipient, and sends along the resulting ciphertext and the hash value to Facebook's servers. Facebook signs the hash and forwards on the whole package — signature, HMAC hash, and ciphertext — to the recipient, who decrypts and checks the validity of the HMAC output using the recovered HMAC key. Should the recipient want to report abuse, their software client sends the signature, message, HMAC hash, and HMAC key to Facebook who can now verify the signature and hash.

While descriptions of Facebook's protocol do not use the term commitment, intuitively that is the role played by HMAC. This may suggest viewing message franking as simply a construction of committing encryption [23]. But committing encryption views the entire ciphertext as the commitment and opens ciphertexts by revealing the secret key. Neither is true of the Facebook scheme.

A new primitive: compactly committing AEAD. We introduce a new cryptographic primitive that captures the properties targeted in verifiable abuse reporting. We refer to it as compactly committing AEAD. This is an AEAD scheme for which a small portion of the ciphertext can be used as a commitment to the message. Decryption reveals an opening for the message, and the scheme comes equipped with an additional verification algorithm that can check the commitment. This formalization has some similarity to one for non-AEAD symmetric encryption due to Gertner and Herzberg [36], but differs in important ways and, in short, their treatment does not suffice for message franking (see Section 9 for more detailed comparisons).

Formalizing security for committing AEAD schemes requires care. Informally we want confidentiality, ciphertext integrity, and that the ciphertexts are binding commitments to their underlying plaintexts. While seemingly a straightforward adaptation of real-or-random style confidentiality and ciphertext integrity notions would suffice [52, 54, 55], this turns out to provide only a weaker form of security in which reporting abuse may invalidate security of the encryption moving forward. In short, this is because the opening might reveal cryptographic key material, e.g., if the secret key is itself used as the opening. We refer to this as single-opening (SO) security. We formalize also multiple-opening (MO) security notions which, in addition to the usual challenge oracles, gives the adversary the ability to obtain regular encryptions and decryptions (which, by our syntax, reveals the opening should a ciphertext be valid) under the target key. Analogously to previous AEAD treatments [55], we formalize this both via an all-in-one security game that simultaneously establishes confidentiality and integrity, and as separate notions for confidentiality and integrity. We prove them equivalent.

Standard integrity notions like INT-CTXT do not by themselves imply that the ciphertext is a binding commitment to the underlying message. We introduce a notion called receiver binding, which is similar to the binding notions from the commitment literature, notions from the robust encryption literature [1, 32, 33], and the prior notion of binding for committing encryption due to Gertner and Herzberg. Importantly, we deal with the fact that only a portion of the ciphertext is committing, and other details such as associated data. Achieving receiver binding means that no computationally limited adversary can find two opening, message pairs that verify for the same committing portion of a ciphertext.

At first glance this seemed like the end of the story with regards to binding security. But in the message franking setting, schemes that are only receiver binding may spectacularly fail to ensure verifiable abuse reporting. In particular, we show how schemes that are receiver binding still allow the following attack: a sender carefully chooses a ciphertext so that an abusive message is correctly decrypted by the receiver, but verification with the resulting opening of that message fails. Such an attack is devastating and arises quickly without careful design. We give an example of a natural performance improvement for the Facebook scheme that provably enjoys confidentiality, ciphertext integrity, and receiver binding, yet subtly falls to this attack. We therefore formalize and target meeting a *sender* binding property that rules out such attacks.

Legacy schemes. With formal notions in place, we start by investigating whether existing, in-use AEAD schemes are compactly committing. For these legacy schemes the opening is taken to be the secret key and per-message randomness used, and in each case we identify a small portion of the ciphertext to take as the committing portion. In this context proving receiver binding also proves the scheme to be committing in the more traditional sense.

As mentioned, AEAD schemes are not in general binding via simple counterexamples. We therefore analyze specific constructions, focusing on three important schemes. The first, Encode-then-Encipher [12], uses a variable-input-length tweakable block cipher to build an authenticated encryption scheme by padding messages with randomness and redundancy information (zero bits). We show that, modeling the underlying tweakable cipher as ideal, one can show that taking a security-parameter number of bits of the ciphertext as the commitment is both receiver and sender binding. Verification re-encrypts the message and checks that the resulting ciphertext properly matches the commitment value.

We next investigate Encrypt-then-MAC constructions [9], which are particularly relevant here given that Signal [44], and in turn Facebook messenger, uses AES-CBC followed by HMAC for authenticated encryption of messages. In practice, one uses a key-derivation function to derive an encryption key and a MAC key. Interestingly, if one uses as opening those two separate keys, then a simple attack shows that this scheme is *not* receiver binding. If, however, one uses the input to the KDF as the opening, we can prove receiver binding assuming the KDF and MAC are collision resistant. Notably this rules out using CMAC [41], PMAC [18], and Carter-Wegman MACs [59], but Encrypt-then-HMAC suffices.

This means that in Facebook messenger the underlying encryption already suffices as a single-opening-secure committing AEAD scheme. Moreover, due to ratcheting [14, 26, 45] Signal never reuses a symmetric key. Thus Facebook could have avoided the dedicated HMAC commitment. Admittedly they may be uncomfortable — for reason of psychological acceptability — with an architecture that sends decryption keys to Facebook despite the fact that this represents no harm to future or past communications.

We finally investigate MAC-then-Encrypt, the mode of operation underlying TLS 1.2 and before. The binding properties of MAC-then-Encrypt were briefly investigated in a recent paper that used TLS 1.2 records as commitments [58], including a brief proof sketch of receiver binding when taking the entire ciphertext as the commitment. We expand on their proof sketch and provide a full proof for the scheme instantiated with CBC-mode and HMAC (the instantiation used in TLS), taking a small constant number of ciphertext blocks as the committing portion. Interestingly this proof, unlike that of Encrypt-then-MAC, required modeling the block cipher underlying CBC-mode as an ideal cipher and HMAC as a random oracle [28].

Commit-then-Encrypt constructions. We next turn to analyzing generic constructions that combine a commitment with an existing AE scheme. We provide a generalization of the Facebook scheme, and show that it is multiple-opening secure and both sender and receiver binding, assuming only that the underlying AEAD scheme is sound and the commitment is unique. HMAC is a unique commitment, thereby giving us the first formal security analysis of Facebook's message franking scheme. One can also use a non-malleable commitment [29]. If one instead uses a malleable commitment, then the scheme will not achieve ciphertext integrity.

We also offer an alternative composition that removes the need for non-malleable commitments, and also can improve performance in the case that associated data is relatively long. Briefly, we use a commitment to the associated data and message as the associated data for the underlying AEAD scheme. This indirectly binds the encryption ciphertext to the associated data, without paying the cost of twice processing it.

Both these constructions are multiple-opening secure, since the commitment opening is independent of the underlying AE keys. This is intuitively simple but the proof requires care — commitments play a role in achieving CTXT and so we must take care to show that unopened encryptions, despite using the same keys as opened encryptions, retain ciphertext integrity. See the body for details.

The Committing Encrypt-and-PRF (CEP) scheme. The generic constructions that meet multiple-opening security are slower than existing (single-opening secure) AEAD schemes, since they require an additional cryptographic pass over the message. This represents approximately a 1.5x slowdown both for encryption and decryption. For the expected workload in messaging applications that consists primarily of relatively short plaintexts, this may not matter, but if one wants to use committing AEAD for large plaintexts such as image and video attachments or in streaming settings (e.g., a committing version of TLS) the overhead will add up quickly.

We therefore offer a new AEAD scheme, called Committing Encrypt-and-PRF (CEP) that simultaneously enjoys multiple-opening security while also retaining the two-pass performance of standard AEAD schemes. As an additional bonus we make the scheme nonce-based [54], meaning that it is derandomized and only needs to be used with non-repeating nonces. (We formalize nonce-based committing AEAD in the body; it is largely similar as the randomized variant.)

The basic idea is to adapt an Encrypt-and-PRF style construction to be compactly committing and multiple-opening. To do so we derive one-time use

PRF keys from the nonce, and compute a tag that is two-part. The commitment value for the ciphertext is the output of a keyed hash that is simultaneously a PRF when the key is private and collision resistant when it is adversarially chosen. The latter is critical since receiver binding requires, in this context, a collision-resistance property. If one stopped here, then the scheme would not be secure, since openings reveal the PRF's key, rendering it only CR, and CR is not enough to prevent future ciphertext forgeries. We therefore additionally run a one-time PRF (with key that is never opened) over this commitment value to generate a tag that is also checked during decryption. Ultimately we prove that the scheme achieves our notions of sender binding, receiver binding, and multiple-opening confidentiality and ciphertext integrity.

We strove to make the scheme simple and fast. Instantiated with a stream cipher such as AES-CTR-mode or ChaCha20, we require just a single secret key and use the stream cipher to generate not only the one-time keys for the PRFs but also a pad for encrypting the message. Because we need a collision-resistant PRF, our suggested instantiation is HMAC, though other multi-property hash functions [10] would work as well.

Future directions. Our work has focused on the symmetric encryption portion of messaging protocols, but one can also ask how the landscape changes if one holistically investigates the public-key protocols or key exchange in particular. Another important direction is to understand the potential tension between committing AEAD and security in the face of selective opening attacks (SOA) [7,8]. Our current definitions do not model SOAs. (An SOA would allow, for example, a compromise of the full cryptographic key, not just the ability to get openings.) While it may seem that committing encryption and SOA security are at odds, we actually conjecture that this is not fundamental (particularly in the random oracle model), and future work will be able to show SOA-secure compactly committing AEAD.

2 Preliminaries

We fix some alphabet Σ, e.g., $\Sigma = \{0,1\}$. For any $x \in \Sigma^*$ let $|x|$ denote its length. We write $x \leftarrow\!\!\$\, X$ to denote uniformly sampling from a set X. We write $X \,\|\, Y$ to denote concatenation of two strings. For a string X of n bits, we will write $X[i,\ldots,j]$ for $i < j \leq n$ to mean the substring of X beginning at index i and ending at index j. For notational simplicity, we assume that one can unambiguously parse $Z = X \,\|\, Y$ into its two parts, even for strings of varying length. For strings $X, Y \in \{0,1\}^*$ we write $X \oplus Y$ to denote taking the XOR of $X[1,\ldots,\min\{|X|,|Y|\}] \oplus Y[1,\ldots,\min\{|X|,|Y|\}]$.

We use code-based games [13] to formalize security notions. A game G is a sequence of pseudocode statements, with variables whose type will be clear from context. Variables are implicitly initialized to appropriate defaults for their type (zero for integers, empty set for sets, etc.). Each variable is a random variable in the probability distribution defined by the random coins used to execute the game. We write $\Pr[G \Rightarrow y]$ to denote the event that the game outputs a value y.

Associated to this pseudocode is some fixed RAM model of computation where most operations are unit cost. We will use "big-O" notation $\mathcal{O}(\cdot)$ to hide only small constants that do not materially impact the interpretation of our results.

We will work in the random oracle model (ROM) [11] and the ideal cipher model (ICM). In the ROM, algorithms and adversaries are equipped with an oracle that associates to each input a random output of some length that will vary by, and be clear from, context. In the ICM, algorithms and adversaries are equipped with a pair of oracles. The first takes input a key, a tweak, and a message, all bit strings of some lengths k, t, and n, respectively. Each key, tweak pair selects a random permutation on $\{0,1\}^n$. The second oracle takes as input a key, a tweak, and an n-bit value, and returns the inverse of the permutation selected by the key and tweak applied to the value.

Below, we will only discuss the time complexity of a reduction if bounding it is non-trivial. Otherwise we will omit discussions of time complexity.

Symmetric encryption. A nonce-based authenticated encryption (AE) scheme $\mathsf{SE} = (\mathsf{Kg}, \mathsf{enc}, \mathsf{dec})$ consists of a triple of algorithms. Associated to it are a key space $\mathcal{K} \subseteq \Sigma^*$, nonce space $\mathcal{N} \subseteq \Sigma^*$, header space $\mathcal{H} \subseteq \Sigma^*$, message space $\mathcal{M} \subseteq \Sigma^*$, and ciphertext space $\mathcal{C} \subseteq \Sigma^*$. The randomized key generation algorithm Kg outputs a secret key $K \in \mathcal{K}$. Canonically Kg selects $K \leftarrow\!\!\$\, \mathcal{K}$ and outputs K. Encryption enc is deterministic and takes as input a four-tuple $(K, N, H, M) \in (\Sigma^*)^4$ and outputs a ciphertext C or a distinguished error symbol $\perp$. We require that $\mathsf{enc}(K, N, H, M) \neq \perp$ if $(K, N, H, M) \in \mathcal{K} \times \mathcal{N} \times \mathcal{H} \times \mathcal{M}$. Decryption dec is deterministic and takes as input a tuple $(K, N, H, C) \in (\Sigma^*)^4$ and outputs a message M or $\perp$.

An SE scheme is correct if for any $(K, N, H, M) \in \mathcal{K} \times \mathcal{N} \times \mathcal{H} \times \mathcal{M}$ it holds that $\mathsf{dec}(K, N, H, \mathsf{enc}(K, N, H, M)) = M$.

Some schemes that we will analyze predate the viewpoint of nonce-based encryption, including generic compositions that utilize CTR or CBC mode. A randomized SE scheme $\mathsf{SE} = (\mathsf{Kg}, \mathsf{enc}, \mathsf{dec})$ is the same as a nonce-based SE scheme except that we omit nonces everywhere, and have enc take an additional input, the coins, that are assumed to be drawn from some coin space $\mathcal{R} \subseteq \sigma^*$. Correctness now is met if for any $(K, H, M, R) \in \mathcal{K} \times \mathcal{H} \times \mathcal{M} \times \mathcal{R}$ it holds that $\mathsf{dec}(K, H, \mathsf{enc}(K, H, M\,; R)) = M$. We will focus on schemes that are public-coin, meaning the ciphertext includes R explicitly. This is true, for example, of CTR or CBC mode encryption. For notational simplicity, we will assume for such schemes that enc outputs R concatenated with the remainder of the ciphertext.

Pseudorandom functions. For a function $F : \mathcal{K} \times \{0,1\}^* \to \{0,1\}^n$ and adversary $\mathcal{A}$ we define the *pseudorandom function* (PRF) advantage of $\mathcal{A}$ to be

$$\mathbf{Adv}_F^{\mathrm{prf}}(\mathcal{A}) = \left| \Pr\left[K \leftarrow\!\!\$\, \mathcal{K} \colon \mathcal{A}^{F(K,\cdot)} = 1 \right] - \Pr\left[R \leftarrow\!\!\$\, \mathbf{Func} \colon \mathcal{A}^{R(\cdot)} = 1 \right] \right| .$$

We define $\mathbf{Func}$ to be the space of all functions that output n bits.[3] Informally, we say the function F is a PRF if $\mathbf{Adv}_F^{\mathrm{prf}}()$ is small for all efficient adversaries.

[3] We are abusing the formalism here by sampling R from an infinite set; we do so for notational consistency and simplicity.

Below we will sometimes refer to the left-hand experiment as the "real world" and the other as the "ideal world".

In proofs it will be convenient to use multi-user PRF security [4]. We define the MU-PRF advantage of an adversary $\mathcal{A}$ to be

$$\mathbf{Adv}_F^{\text{mu-prf}}(\mathcal{A}) = \left|\Pr\left[\,\mathcal{A}^{\overline{F}(\cdot,\cdot)} \Rightarrow 1\,\right] - \Pr\left[\,\mathcal{A}^{\overline{R}(\cdot,\cdot)} \Rightarrow 1\,\right]\right| \;.$$

where $\overline{F}$ on input a key identifier $S \in \{0,1\}^*$ and a message M, checks if there is a key associated to S, and if not chooses a fresh one $K[S] \leftarrow\!\!\$\, \{0,1\}^k$. It then returns $F(K[S], M)$. The oracle $\overline{R}$ on input a key identifier $S \in \{0,1\}^*$ and a message M, checks if there is a random function associated to S, and if not chooses a fresh one $R[S] \leftarrow\!\!\$\, \text{Func}$. It returns $R[S](M)$. Note that MU-PRF security is implied by PRF security via a standard argument.

Collision-resistance. For a function $F : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^n$ and adversary $\mathcal{A}$, define the *collision-resistance* (CR) advantage as

$$\mathbf{Adv}_F^{\text{cr}}(\mathcal{A}) = \Pr\left[\,((x_1,x_2),(x_1',x_2')) \leftarrow\!\!\$\, \mathcal{A} \colon \begin{array}{c} F(x_1,x_2) = F(x_1',x_2'), \\ (x_1,x_2) \neq (x_1',x_2') \end{array}\right] \;.$$

Informally, we say F is collision-resistant if $\mathbf{Adv}_F^{\text{cr}}()$ is small for all efficient adversaries.

Commitment schemes with verification. A commitment scheme with verification $\mathsf{CS} = (\mathsf{Com}, \mathsf{VerC})$ consists of two algorithms. Associated to any commitment scheme is an opening space $\mathcal{K}_f \subseteq \Sigma^*$, a message space $\mathcal{M} \subseteq \Sigma^*$, and a commitment space $\mathcal{C} \subseteq \Sigma^*$. The algorithm Com is randomized and takes as input a $M \in \Sigma^*$ and outputs a pair $(K, C) \in \mathcal{K}_f \times \mathcal{C}$ or an error symbol $\perp$. We assume that Com returns $\perp$ with probability one if $M \notin \mathcal{M}$. The algorithm VerC is deterministic. It takes input a tuple $(K, C, M) \in \Sigma^*$ and outputs a bit. We assume that VerC returns 0 if its input $(K, C, M) \notin \mathcal{K}_f \times \mathcal{C} \times \mathcal{M}$. We assume that the commitment values C are of some fixed length (typically denoted by t).

A commitment scheme (with verification) is correct if for all $M \in \mathcal{M}$ it holds that $\Pr[\mathsf{VerC}(\mathsf{Com}(M), M) = 1] = 1$ where the probability is over the coins used by Com. We will not use the alternate definition of commitments with opening [21]. We can formalize the binding security notion of our commitment scheme as a game. Formally, the game $\text{vBIND}_{\mathsf{CS},\mathcal{A}}$ first runs the adversary $\mathcal{A}$ who outputs a tuple (K_c, M, K_c', M', C). The game then runs $b \leftarrow \mathsf{VerC}(K_c, C, M)$ and $b' \leftarrow \mathsf{VerC}(K_c', C, M')$. The game outputs true if $M \neq M'$ and $b = b' = 1$ and false otherwise. To a commitment scheme CS and adversary $\mathcal{A}$ we associate the vBIND advantage

$$\mathbf{Adv}_{\mathsf{CS}}^{\text{v-bind}}(\mathcal{A}) = \Pr\left[\,\text{vBIND}_{\mathsf{CS},\mathcal{A}} \Rightarrow \mathsf{true}\,\right] \;.$$

The probability is over the coins used by the game.

Commitment schemes should enjoy a hiding property as well. Traditionally this is formalized as a left-or-right indistinguishability notion (q.v., [6]). For our purposes we will target a stronger notion, analogous to real-or-random (ROR) security for symmetric encryption. It asks that a commitment be indistinguishable from a random bit string while the opening remaining secret. Game $\mathrm{ROR1}_{\mathsf{CS},\mathcal{A}}$ runs an adversary $\mathcal{A}$ and gives it access to an oracle **Com** to which it can query messages. The oracle computes $(K_c, C) \leftarrow\!\!\$\ \mathsf{Com}(M)$ and returns C. The adversary outputs a bit, and the game outputs true if the bit is one. Game $\mathrm{ROR0}_{\mathsf{CS},\mathcal{A}}$ is similar except that the oracle returns a string of random bits of length $|C|$ and the game outputs true if the adversary outputs zero. We define the advantage by $\mathbf{Adv}^{\text{cs-ror}}_{\mathsf{CS}}(\mathcal{A}) = |\Pr[\,\mathrm{ROR1}_{\mathsf{CS},\mathcal{A}} \Rightarrow \mathsf{true}\,] - \Pr[\,\mathrm{ROR0}_{\mathsf{CS},\mathcal{A}} \Rightarrow \mathsf{false}\,]|$.

HMAC is a good commitment. Any PRF that is also collision-resistant meets our security goals for commitments. In particular, one can build a commitment scheme $\mathsf{CS}[F] = (\mathsf{Com}, \mathsf{VerC})$ works from any function $F\colon \mathcal{K} \times \{0,1\}^* \to \{0,1\}^n$ as follows. Commitment $\mathsf{Com}(M)$ chooses a fresh value $K \leftarrow\!\!\$\ \mathcal{K}$, computes $C \leftarrow F(K, M)$ and outputs (K, C). Verification $\mathsf{VerC}(K, C, M)$ outputs one if $F(K, M) = C$ and zero otherwise. Then the following theorem captures the security of this commitment scheme, which rests on the collision resistance and PRF security of F. A proof of this theorem appears in the full version of the paper.

Theorem 1. *Let F be a function and $\mathsf{CS}[F]$ be the commitment scheme built from it as described above. Then for any efficient adversaries $\mathcal{A}$ making at most q queries in game ROR and $\mathcal{A}'$ in game* vBIND *respectively, there exists a pair of adversaries $\mathcal{B}, \mathcal{B}'$ so that*

$$\mathbf{Adv}^{\text{cs-ror}}_{\mathsf{CS}[F]}(\mathcal{A}) \le q \cdot \mathbf{Adv}^{\text{prf}}_{F}(\mathcal{B}) \quad \text{and} \quad \mathbf{Adv}^{\text{v-bind}}_{\mathsf{CS}[F]}(\mathcal{A}') \le \mathbf{Adv}^{\text{cr}}_{F}(\mathcal{B}')\ .$$

The adversary $\mathcal{B}$ runs in time that of $\mathcal{A}$ and makes the same number of oracle queries as $\mathcal{A}$. Adversary $\mathcal{B}'$ runs in time that of $\mathcal{A}'$.

As the underyling function needs to be both CR and a good PRF, a suitable candidate would be HMAC [5], i.e., $F(K, M) = \mathsf{HMAC}(K, M)$. Other multi-property hash functions [10] could be used as well. The Facebook franking scheme (discussed in Section 3) uses a non-standard HMAC-based commitment based on $F(K, M) = \mathsf{HMAC}(K, M \,\|\, K)$. We will assume HMAC remains a PRF when used in this non-standard way. One can substantiate this assumption directly in the random oracle model, or using techniques from the key-dependent message literature [19, 37].

3 Message Franking and End-to-End Encryption

In end-to-end encrypted messaging services there exists a tension between message privacy and reporting abusive message contents to service providers. The latter is important to flag abusive accounts, but reports need to be verifiable, meaning that the provider can check the contents of the allegedly abusive message *and* be certain that it was the message sent. Otherwise abuse-reporting mechanisms could themselves be abused to make false accusations.

A recipient can send the allegedly abusive plaintext to the service provider, but message privacy guarantees that the provider does not know whether the

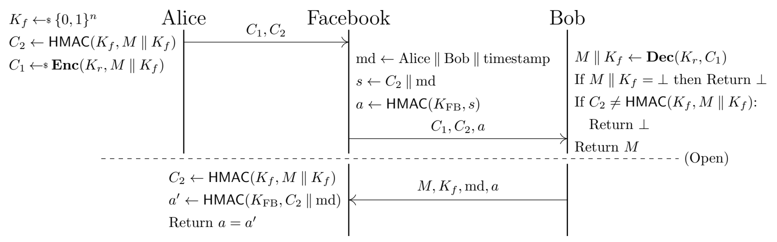
$K_f \leftarrow\!\!\$\ \{0,1\}^n$
Alice
Facebook
Bob
$C_2 \leftarrow \mathsf{HMAC}(K_f, M \parallel K_f)$
C_1, C_2
$C_1 \leftarrow\!\!\$\ \mathbf{Enc}(K_r, M \parallel K_f)$
md ← Alice ∥ Bob ∥ timestamp
$s \leftarrow C_2 \parallel \mathrm{md}$
$a \leftarrow \mathsf{HMAC}(K_{\mathrm{FB}}, s)$
C_1, C_2, a
$M \parallel K_f \leftarrow \mathbf{Dec}(K_r, C_1)$
If $M \parallel K_f = \perp$ then Return $\perp$
If $C_2 \neq \mathsf{HMAC}(K_f, M \parallel K_f)$:
Return $\perp$
Return M
(Open)
$C_2 \leftarrow \mathsf{HMAC}(K_f, M \parallel K_f)$
$a' \leftarrow \mathsf{HMAC}(K_{\mathrm{FB}}, C_2 \parallel \mathrm{md})$
Return $a = a'$
M, K_f, md, a

Fig. 2. Facebook's message franking protocol [47]. The key K_r is a one-time-use symmetric key derived as part of the record layer protocol. The top portion is the sending of an encrypted message to the recipient. The bottom portion is the abuse reporting protocol.

alleged message was in fact the one sent.[4] A seeming solution would be for the service to log ciphertexts, and have the recipient disclose the secret key to allow the provider to decrypt the ciphertext. Not only is this impractical due to the storage requirements, but it also does *not* guarantee that the decrypted message is correct. It could be that the recipient chose a key that somehow decrypts the (legitimate) ciphertext to a fake message. Ultimately what is required for this to work is for the encryption to be committing: no computationally efficient adversary can find a secret key that decrypts the ciphertext to anything but the originally encrypted message.

Facebook's approach. Facebook recently detailed a new cryptographic mechanism [31,47] targeting verifiable abuse reporting on Facebook messenger, which uses end-to-end encryption based on Signal [61]. The basic idea is to force the sender to provide a commitment, sent in the clear, to the plaintext message. A diagram of Facebook's protocol, that they call "message franking" (as in "speaking frankly"), is shown in Figure 2. The sender first applies HMAC with a fresh key K_f to the concatenation of the message and K_f to produce a value C_2, and then encrypts using an AEAD scheme the message and K_f to produce a ciphertext C_1 using a key K_r shared with the recipient. Then (C_1, C_2) is sent to Facebook. Facebook applies HMAC with its own secret key K_{FB} to C_2 to get a tag a, and sends to the recipient (C_1, C_2, a). The recipient decrypts C_1, recovers the message M and key K_f and checks the value $C_2 = \mathsf{HMAC}(K_f, M \parallel K_f)$. To report abuse, the recipient sends M, K_f, and a to Facebook. Facebook recomputes $\mathsf{HMAC}(K_f, M \parallel K_f)$ and checks the tag a.

It is clear that the sender is using HMAC as a cryptographic commitment to the message. (This terminology is not used in their technical specifications.) The use of HMAC by Facebook to generate the tag a is simply to forego having to store commitments, instead signing them so that they can be outsourced to recipients for storage and verified should an abuse report come in.

[4] Of course, if the recipient is running a trusted client, then this assertion could be trusted. We are concerned with the case that the client is subverted.

There are interesting security issues that could arise with Facebook's scheme, and cryptographic abuse reporting in general, that are orthogonal to the ones discussed here. In particular, binding Facebook's tag to the communicating parties seems crucial: otherwise a malicious party could create a sock-puppet (i.e. fake) account, send itself an abusive message, then accuse a victim of having sent it.

While the design looks reasonable, and the Facebook white paper provides some informal discussion about security, there has been no formal analysis to date. It is also not clear what security properties the main cryptographic construction — combining a commitment with AEAD — should satisfy. We rectify this by introducing, in the following section, the notion of committing AEAD. This will allow us not only to analyze Facebook's franking scheme, but to suggest alternative designs, including ones that are legacy-compatible with existing deployed AEAD schemes and do not, in particular, require adding an additional dedicated commitment.

4 Committing AEAD

Formally, a committing AEAD scheme $\mathsf{CE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Ver})$ is a four-tuple of algorithms. Associated to a scheme is a key space $\mathcal{K} \subseteq \Sigma^*$, header space $\mathcal{H} \subseteq \Sigma^*$, message space $\mathcal{M} \subseteq \Sigma^*$, ciphertext space $\mathcal{C} \subseteq \Sigma^*$, opening space $\mathcal{K}_f \subseteq \Sigma^*$, and franking tag space $\mathcal{T} \subseteq \Sigma^*$.

- ***Key generation***: The randomized key generation algorithm Kg outputs a secret key $K \in \mathcal{K}$. We write $K \leftarrow\!\!\$\ \mathsf{Kg}$ to denote executing key generation.
- ***Encryption***: Encryption Enc is randomized. The input to encryption is a triple $(K, H, M) \in (\Sigma^*)^3$ and the output is a pair $(C_1, C_2) \in \mathcal{C} \times \mathcal{T}$ or a distinguished error symbol $\perp$. Unlike with regular symmetric encryption, the output includes two components: a ciphertext C_1 and a franking tag C_2. We also refer to C_2 as the commitment. We require that $\mathsf{Enc}(K, H, M) \neq \perp$ if $(K, H, M) \in \mathcal{K} \times \mathcal{H} \times \mathcal{M}$. We write $(C_1, C_2) \leftarrow\!\!\$\ \mathsf{Enc}(K, H, M)$ to denote executing encryption.
- ***Decryption***: Decryption, which is deterministic, takes as input a tuple $(K, H, C_1, C_2) \in (\Sigma^*)^4$ and outputs a message, opening value pair $(M, K_f) \in \mathcal{M} \times \mathcal{K}_f$ or $\perp$. We write $(M, K_f) \leftarrow \mathsf{Dec}(K, H, C_1, C_2)$ to denote executing decryption.
- ***Verification***: Verification, which is deterministic, takes as input a tuple $(H, M, K_f, C_2) \in (\Sigma^*)^4$ and outputs a bit. For $(H, M, K_f, C_2) \notin \mathcal{H} \times \mathcal{M} \times \mathcal{K}_f \times \mathcal{T}$, we assume that Ver outputs 0. We write $b \leftarrow \mathsf{Ver}(H, M, K_f, C_2)$ to denote executing verification.

We will often place K in the subscript of relevant algorithms. For example, $\mathsf{Enc}_K(H, M) = \mathsf{Enc}(K, H, M)$ and $\mathsf{Dec}_K(H, C_1, C_2) = \mathsf{Dec}(K, H, C_1, C_2)$.

We require that CE schemes output ciphertexts whose lengths are determined solely by the length of the header and message. Formally this means that there exists a function $\mathsf{clen} : \mathbb{N} \times \mathbb{N} \to \mathbb{N} \times \mathbb{N}$ such that for all $(K, H, M) \in \mathcal{K} \times \mathcal{H} \times \mathcal{M}$ it holds that $\Pr[(|C_1|, |C_2|) = \mathsf{clen}(|H|, |M|)] = 1$ where $(C_1, C_2) \leftarrow\$ \mathsf{Enc}_K(H, M)$ and the probability is over the coins used by encryption.

We say a CE scheme has *decryption correctness* if for all $(K, H, M) \in \mathcal{K} \times \mathcal{H} \times \mathcal{M}$ it holds that $\Pr[\mathsf{Dec}(K, H, C_1, C_2) = M] = 1$ where the probability is taken over the coins used to compute $(C_1, C_2) \leftarrow\$ \mathsf{Enc}(K, H, M)$.

We say that a scheme has *commitment correctness* if for all $(K, H, M) \in \mathcal{K} \times \mathcal{H} \times \mathcal{M}$ it holds that $\Pr[\mathsf{Ver}(H, M, K_f, C_2) = 1] = 1$ where the probability is taken over the random variables used in the experiment

$$(C_1, C_2) \leftarrow\$ \mathsf{Enc}_K(H, M)\ ;\ (M, K_f) \leftarrow \mathsf{Dec}_K(H, C_1, C_2)\ ;\ \text{Return } (K_f, C_2)$$

Our formulation of CE schemes is a generalization of that for conventional (randomized) AE schemes in the following sense. One can consider an AE scheme as a CE scheme that has encryption output the entire ciphertext as C_2, decryption output an empty string for the opening value, and has verify always return one.

Compactly committing AEAD. In our formalism, a ciphertext has two components. A scheme may output $C_1 = \varepsilon$ and a C_2 value that therefore consists of the entire ciphertext. This embodies the traditional viewpoint on committing AEAD, in which the entire ciphertext is viewed as the commitment. But we are more general, and in particular our formalism allows schemes with *compact* commitments, by which we mean schemes for which $|C_2|$ is small. In particular we will want $|C_2|$ to be linear in the security-parameter, rather than linear in the message length. One can make any CE scheme compact by hashing the ciphertext with a collision-resistant (CR) hash function, as we show formally in a moment. But we will also show compact schemes that have better performance.

Single versus multiple openings. In some protocols, we may wish to use a CE scheme so that multiple different ciphertexts, encrypted under the same secret key, can be opened without endangering the privacy or integrity of other unopened ciphertexts. In other contexts, the CE scheme's opening need only be "single-use" — the secret key will not continue to be used after an opening. An example of the latter is Signal, which due to ratcheting effectively has a fresh secret key per message. As we will now discuss, whether one wants single-opening or multiple-opening CE must be reflected in the security definitions.

Confidentiality. We want our CE schemes to provide message confidentiality. We will in fact adapt the stronger real-or-random notion from the AE literature (q.v., [55]) to CE. At a high level we ask that no adversary can distinguish between legitimate CE encryptions and (pairs of) random bit strings. A complexity arises in the multi-opening case, where we want confidentiality to hold even after openings occur. We handle this by giving the attacker an additional pair of oracles, one for encryption and decryption. We must take care to avoid trivial wins, of course, separating use of the real oracles from the challenge ones. We also additionally require that the adversary can only query its decryption oracle on valid ciphertexts returned from the encryption oracle. This all is formalized in the games MO-REAL$_{\mathsf{CE},\mathcal{A}}$ and MO-RAND$_{\mathsf{CE},\mathcal{A}}$ shown in Figure 3. We measure

```
$\text{MO-REAL}_{\mathsf{CE},\mathcal{A}}$:
$K \leftarrow\!\!\$\ \mathsf{Kg}$
$b' \leftarrow\!\!\$\ \mathcal{A}^{\mathbf{Enc},\mathbf{Dec},\mathbf{ChalEnc}}$
Return $b'$

$\mathbf{Enc}(H, M)$
$(C_1, C_2) \leftarrow\!\!\$\ \mathsf{Enc}_K(H, M)$
$\mathcal{Y}_1 \leftarrow \mathcal{Y}_1 \cup \{(H, C_1, C_2)\}$
Return $(C_1, C_2)$

$\mathbf{Dec}(H, C_1, C_2)$
If $(H, C_1, C_2) \notin \mathcal{Y}_1$ then
    Return $\bot$
$(M, K_f) \leftarrow \mathsf{Dec}_K(H, C_1, C_2)$
Return $(M, K_f)$

$\mathbf{ChalEnc}(H, M)$
$(C_1, C_2) \leftarrow\!\!\$\ \mathsf{Enc}_K(H, M)$
Return $(C_1, C_2)$
```

```
$\text{MO-RAND}_{\mathsf{CE},\mathcal{A}}$:
$K \leftarrow\!\!\$\ \mathsf{Kg}$
$b' \leftarrow\!\!\$\ \mathcal{A}^{\mathbf{Enc},\mathbf{Dec},\mathbf{ChalEnc}}$
Return $b'$

$\mathbf{Enc}(H, M)$
$(C_1, C_2) \leftarrow\!\!\$\ \mathsf{Enc}_K(H, M)$
$\mathcal{Y}_1 \leftarrow \mathcal{Y}_1 \cup \{(H, C_1, C_2)\}$
Return $(C_1, C_2)$

$\mathbf{Dec}(H, C_1, C_2)$
If $(H, C_1, C_2) \notin \mathcal{Y}_1$ then
    Return $\bot$
$(M, K_f) \leftarrow \mathsf{Dec}_K(H, C_1, C_2)$
Return $(M, K_f)$

$\mathbf{ChalEnc}(H, M)$
$(\ell_1, \ell_2) \leftarrow \mathsf{clen}(|H|, |M|)$
$(C_1, C_2) \leftarrow\!\!\$\ \{0,1\}^{\ell_1} \times \{0,1\}^{\ell_2}$
Return $(C_1, C_2)$
```

```
$\text{MO-CTXT}_{\mathsf{CE},\mathcal{A}}$:
$K \leftarrow\!\!\$\ \mathsf{Kg}$ ; $\mathsf{win} \leftarrow \mathsf{false}$
$\mathcal{A}^{\mathbf{Enc},\mathbf{Dec},\mathbf{ChalDec}}$
Return $\mathsf{win}$

$\mathbf{Enc}(H, M)$
$(C_1, C_2) \leftarrow\!\!\$\ \mathsf{Enc}_K(H, M)$
$\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(H, C_1, C_2)\}$
Return $(C_1, C_2)$

$\mathbf{Dec}(H, C_1, C_2)$
Return $\mathsf{Dec}_K(H, C_1, C_2)$

$\mathbf{ChalDec}(H, C_1, C_2)$
If $(H, C_1, C_2) \in \mathcal{Y}$ then
    Return $\bot$
$(M, K_f) \leftarrow \mathsf{Dec}_K(H, C_1, C_2)$
If $M \neq \bot$ then
    $\mathsf{win} \leftarrow \mathsf{true}$
Return $(M, K_f)$
```

Fig. 3. Confidentiality (left two games) and ciphertext integrity (rightmost) games for committing AEAD.

the multiple-openings real-or-random (MO-ROR) advantage of an adversary $\mathcal{A}$ against a scheme CE by

$$\mathbf{Adv}^{\text{mo-ror}}_{\mathsf{CE}}(\mathcal{A}) = |\Pr[\,\text{MO-REAL}_{\mathsf{CE},\mathcal{A}} \Rightarrow 1\,] - \Pr[\,\text{MO-RAND}_{\mathsf{CE},\mathcal{A}} \Rightarrow 1\,]| \ .$$

The single-opening ROR (SO-ROR) games $\text{REAL}_{\mathsf{CE},\mathcal{A}}$ and $\text{RAND}_{\mathsf{CE},\mathcal{A}}$ are identical to $\text{MO-REAL}_{\mathsf{CE},\mathcal{A}}$ and $\text{MO-RAND}_{\mathsf{CE},\mathcal{A}}$ in Figure 3 except that we omit the **Enc** and **Dec** oracles. We measure the single-openings real-or-random (ROR) advantage of an adversary $\mathcal{A}$ against a scheme CE by

$$\mathbf{Adv}^{\text{ror}}_{\mathsf{CE}}(\mathcal{A}) = |\Pr[\,\text{REAL}_{\mathsf{CE},\mathcal{A}} \Rightarrow 1\,] - \Pr[\,\text{RAND}_{\mathsf{CE},\mathcal{A}} \Rightarrow 1\,]| \ .$$

Ciphertext integrity. We also want our CE schemes to enjoy ciphertext integrity. As with confidentiality, we will lift the standard (randomized) AEAD security notions to the multiple-opening and single-opening CE settings. The game $\text{MO-CTXT}_{\mathsf{CE},\mathcal{A}}$ is shown in Figure 3. The adversary can obtain encryptions and decryptions under the secret key, and its goal is to query a valid ciphertext to a challenge decryption oracle. That ciphertext must not have been returned by the encryption oracle. We measure the multiple-openings ciphertext integrity (MO-CTXT) advantage of an adversary $\mathcal{A}$ against a scheme CE by

$$\mathbf{Adv}^{\text{mo-ctxt}}_{\mathsf{CE}}(\mathcal{A}) = \Pr[\,\text{MO-CTXT}_{\mathsf{CE},\mathcal{A}} \Rightarrow \mathsf{true}\,] \ .$$

As with confidentiality, we can also specify a single-opening version of security by removing the decryption oracle **Dec** from game $\text{MO-CTXT}_{\mathsf{CE},\mathcal{A}}$. Let the resulting game be $\text{CTXT}_{\mathsf{CE},\mathcal{A}}$. We measure the single-openings ciphertext integrity

```
s-BIND$^{\mathcal{A}}_{\mathsf{CE}}$:
$(K, H, C_1, C_2) \leftarrow\$ \mathcal{A}$
$(M', K_f) \leftarrow \mathsf{Dec}(K, H, C_1, C_2)$
If $M' = \bot$ then Return false
$b \leftarrow \mathsf{Ver}(H, M', K_f, C_2)$
If $b = 0$ then
    Return true
Return false
```

```
r-BIND$^{\mathcal{A}}_{\mathsf{CE}}$:
$((H, M, K_f), (H', M', K'_f), C_2) \leftarrow\$ \mathcal{A}$
$b \leftarrow \mathsf{Ver}(H, M, K_f, C_2)$
$b' \leftarrow \mathsf{Ver}(H', M', K'_f, C_2)$
If $(H, M) = (H', M')$ then
    Return false
Return $(b = b' = 1)$
```

Fig. 4. Binding security games for committing AEAD. Sender binding (left game) models a setting where a malicious sender wants to send a message, but prevent commitment opening from succeeding. Receiver binding (right game) models a setting where a sender and recipient collude to open a ciphertext to different messages.

(CTXT) advantage of an adversary $\mathcal{A}$ against a scheme CE by

$$\mathbf{Adv}^{\mathrm{ctxt}}_{\mathsf{CE}}(\mathcal{A}) = \Pr\left[\, \mathrm{CTXT}_{\mathsf{CE},\mathcal{A}} \Rightarrow \mathsf{true} \,\right] .$$

All-in-one notions. We have given separate confidentiality and ciphertext integrity notions. As with traditional AEAD security, however, we can alternatively give an all-in-one notion that simultaneously captures confidentiality and integrity goals. We defer the details to the full version of this work.

Security for AEAD. Given the fact that CE schemes encompass (randomized) AEAD schemes as well (see our comments above), we note that the ROR and CTXT notions apply to standard (randomized) AE schemes. As a slight abuse of notation, we will therefore use ROR and CTXT and their associated games and advantage measures for the security of traditional AE schemes.

Binding security notions. We introduce two security notions for binding: *sender binding* and *receiver binding*. Sender binding ensures the sender of a message is bound to the message it actually sent. In abuse-reporting scenarios, this prevents the sender of an abusive message from generating a bogus commitment that does not give the receiver the ability to report the message. The pseudocode game s-BIND on the left-hand-side of Figure 4 formalizes this requirement. To an adversary $\mathcal{A}$ and CE scheme CE we associate the "sender binding" advantage

$$\mathbf{Adv}^{\text{s-bind}}_{\mathsf{CE}}(\mathcal{A}) = \Pr\left[\, \text{s-BIND}_{\mathsf{CE},\mathcal{A}} \Rightarrow \mathsf{true} \,\right] .$$

The probability is over the coins used in the game.

A CE scheme can generically meet sender binding by running Ver during Dec and having Dec return $\bot$ if Ver returns 0. We omit the proof of this, which follows by inspection. But legacy AEAD schemes do not do this, and one needs to check sender binding. For new schemes we will see more efficient ways to achieve sender binding.

The second security notion, receiver binding, is a lifting of the more traditional binding notion from commitment schemes (see Section 2). This definition is important in abuse reporting, where it formalizes the intuition that a malicious receiver should not be able to accuse a non-abusive sender of having said something abusive. A malicious receiver could do this by opening one of the sender's ciphertexts to an abusive message instead of the one the sender intended.

The pseudocode game r-BIND is shown on the right in Figure 4. It has an adversary output a pair of triples containing associated data, a message, and an opening. The adversary outputs a franking tag C_2 as well. The adversary wins if verification succeeds on both triples with C_2 and the header/message pairs differ. To a CE scheme CE and adversary $\mathcal{A}$ we associate the "receiver binding" advantage

$$\mathbf{Adv}^{\text{r-bind}}_{\mathsf{CE}}(\mathcal{A}) = \Pr\left[\, \text{r-BIND}_{\mathsf{CE},\mathcal{A}} \Rightarrow \mathsf{true} \,\right] \ .$$

The probability is over the coins used in the game.

It is important to note that r-BIND security does not imply s-BIND security. These notions are, in fact, orthogonal. Moreover, our MO-ROR and MO-CTXT notions do not generically imply either of the binding notions.

Discussion. Our definitions also allow associated data, sometimes referred to as headers. This puts committing AEAD on equal footing with modern authenticated encryption with associated data (AEAD) schemes [52], which require it. That said, modern AEAD schemes are most often formalized as nonce-based, meaning that instead of allowing internal randomness, a non-repeating value (the nonce) is an explicit input and encryption is deterministic. Existing systems relevant to abuse complaints use randomized AEAD (e.g., Signal [44]) that do not meet nonce-based AEAD security. That said, we will explore nonce-based committing AEAD in Section 7.

5 Are Existing AEAD Schemes Committing?

In this section we will study whether existing AEAD schemes meet our security goals for CE. We believe it is important to study legacy schemes for several reasons. If existing AEAD schemes are also committing, it will have important positive and negative implications for deployed protocols (such as OTR or Facebook's franking scheme) that implicitly rely on binding (or non-binding) properties of symmetric encryption. It is also helpful for protocol designers who may want to build a protocol on top of existing legacy encryption. If well-tested, mature implementations of AEAD can be used as CE schemes without code changes, the attack surface of new protocol implementations is minimized.

In this section we only examine the binding properties of schemes, since past work has shown they meet standard definitions for confidentiality and integrity. We will prove that encode-then-encipher and encrypt-then-MAC (EtM) satisfy our binding notions in the ideal cipher model, with the additional requirement that the MAC used in EtM is a collision-resistant PRF. We will prove MAC-then-encrypt meets our binding notions in the random oracle and ideal cipher model. We will also show simple attacks that break binding for real-world modes using Carter-Wegman MACs (GCM and ChaCha20/Poly1305).

```
Enc(K, H, M):
    $R \leftarrow_{\$} \{0,1\}^r$
    $C \leftarrow \widetilde{E}^H_K(M \parallel R \parallel 0^s)$
    $\ell \leftarrow l + r + s - t$
    $C_1 \leftarrow C[1, \ldots, \ell]$
    $C_2 \leftarrow C[\ell + 1, \ldots, l + r + s]$
    Return $(C_1, C_2)$
```

```
Dec(K, H, C_1, C_2):
    $M' \parallel R' \parallel Z \leftarrow \widetilde{D}^H_K(C_1 \parallel C_2)$
    If $Z \neq 0^s$ then
        Return $\perp$
    Return $(M', (R', K))$
```

```
Ver(H, M, K_f, C_2):
    $R \parallel K \leftarrow K_f$
    $\ell \leftarrow l + r + s - t$
    $C \leftarrow \widetilde{E}^H_K(M \parallel R \parallel 0^s)$
    Return $C[\ell{+}1, \ldots, l{+}r{+}s] = C_2$
```

Fig. 5. Encode-then-encipher as a committing AEAD scheme where the commitment is the final t bits of the ciphertext. $\widetilde{E}^H$ and $\widetilde{D}^H$ refer to encryption and decryption for a tweakable blockcipher where the header H is the tweak.

5.1 Committing Encode-then-Encipher

The Encode-then-Encipher (EtE) construction of Bellare and Rogaway shows how to achieve AE security for messages given only a variable-input-length PRP [12]. Their construction is quite simple: given a key $K \in \mathcal{K}$, encrypt a message $M \in \mathcal{M}$ ($|\mathcal{M}| = 2^l$) with header $H \in \mathcal{H}$ by first drawing a random string $R \leftarrow_{\$} \{0,1\}^r$ and computing $c = \widetilde{E}^H_K(M \parallel R \parallel 0^s)$ where $\widetilde{E}^H$ is a tweakable, variable-input length cipher with the header as the tweak. Decrypting a ciphertext M works by first running $M' = \widetilde{D}^H_K(C)$ and checking whether the last s bits of M' are all zero. If they are, we call the message "valid" and output M, else we output $\perp$. For compactness, we commit to only the last t bits of the ciphertext. We must include the randomness used to encrypt in the opening of the commitment. Detailed pseudocode is given in Figure 5. We will assume that E is an ideal tweakable cipher in our proof of r-BIND security.

Theorem 2. *Let* EtE[E] *be the scheme defined above using an ideal tweakable cipher E and parameters $s, t > 0$. Let $\mathcal{A}$ be any adversary making at most q queries to its ideal cipher oracles. Then* $\mathbf{Adv}^{\text{r-bind}}_{\text{EtE}}(\mathcal{A}) \leq \frac{q+1}{2^s} + \frac{q^2}{2^t}$.

The proof will appear in the full version of this work. The scheme achieves perfect s-BIND security: the advantage of any adversary for is zero because the output of decryption is simply re-computed in Ver.

5.2 Encrypt-then-MAC

The classic Encrypt-then-MAC (EtM) construction composes a symmetric encryption scheme and a message authentication code (MAC), by first encrypting the message, then computing the MAC over the ciphertext and any associated data.

Committing EtM. We analyze EtM as a committing AEAD scheme in the case that the encryption and authentication keys are derived via a key derivation function (KDF) that is a collision-resistant pseudorandom function. The scheme EtM[KDF, F, SE] is detailed in Figure 6. Beyond the functions F and KDF, the scheme also makes use of a public-coin randomized symmetric encryption algorithm SE = (Kg, enc, dec) that does not use associated data and whose key

```
Enc(K, H, M):
  $K^e \leftarrow \mathsf{KDF}_K(0)$
  $K^m \leftarrow \mathsf{KDF}_K(1)$
  $R \leftarrow\$ \mathcal{R}$
  $R \,\|\, C \leftarrow \mathsf{enc}_{K^e}(M\ ;\ R)$
  $T \leftarrow F_{K^m}(H \,\|\, R \,\|\, C)$
  Return $(R \,\|\, C, T)$
```

```
Dec(K, H, C_1, C_2):
  $R \,\|\, C \leftarrow C_1$
  $K^e \leftarrow \mathsf{KDF}_K(0)$
  $K^m \leftarrow \mathsf{KDF}_K(1)$
  $T' \leftarrow F_{K^m}(H \,\|\, R \,\|\, C_1)$
  If $T' \neq C_2$ then Return $\bot$
  $M \leftarrow \mathsf{dec}_{K^e}(C_1)$
  If $M = \bot$ then Return $\bot$
  Return $(M, (R, K))$
```

```
Ver(H, M, (R, K), C_2):
  $K^e \leftarrow \mathsf{KDF}_K(0)$
  $K^m \leftarrow \mathsf{KDF}_K(1)$
  $C \leftarrow \mathsf{enc}_{K^e}(M\ ;\ R)$
  $T \leftarrow F_{K^m}(H \,\|\, R \,\|\, C)$
  Return $T = C_2$
```

Fig. 6. Committing AEAD scheme $\mathsf{EtM}[\mathsf{KDF}, F, \mathsf{SE}]$ that composes an encryption scheme $\mathsf{SE} = (\mathsf{Kg}, \mathsf{enc}, \mathsf{dec})$ using random coins from $\mathcal{R}$, a MAC F, and that derives keys via a function KDF.

generation is a random selection of some fixed-length bit string. It is important that the scheme is public coin, as we require the randomness to be recoverable during decryption to be included in the opening.

This scheme arises in practice. The Signal protocol [44], for example, uses HKDF to derive keys for use with CTR mode encryption combined with HMAC. The following theorem proves the committing EtM construction in Figure 6 meets r-BIND if the MAC and key derivation function are both collision-resistant PRFs.

Theorem 3. *Let* $\mathsf{EtM} = \mathsf{EtM}[\mathsf{KDF}, F, \mathsf{SE}]$ *be the EtM construction using functions* F *and* KDF *as well as encryption scheme* SE. *Let* $\mathcal{A}$ *be any* r-BIND$_{\mathsf{EtM}}$ *adversary. Then there exist adversaries* $\mathcal{B}$ *and* $\mathcal{C}$, *each that run in time that of* $\mathcal{A}$, *such that* $\mathbf{Adv}^{\text{r-bind}}_{\mathsf{EtM}}(\mathcal{A}) < \mathbf{Adv}^{\text{cr}}_{F}(\mathcal{B}) + \mathbf{Adv}^{\text{cr}}_{\mathsf{KDF}}(\mathcal{C})$.

The proof of this theorem will appear in the full version of this work. The s-BIND security of $\mathsf{EtM}[\mathsf{KDF}, F, \mathsf{SE}]$ is perfect because verification re-encrypts the plaintext to check the tag.

Two-key EtM is not binding. The use of a KDF to derive the encryption and MAC keys above is requisite to achieve receiver binding security. Consider omitting the KDF steps, and instead letting keys be a pair (K^e, K^m) where each component is chosen randomly. The opening output by encryption and used by verification is instead $(R, (K^e, K^m))$. The rest of the scheme remains the same as that in Figure 6. But it is easy to break the receiver binding for this two-key variant: simply have an adversary $\mathcal{A}$ that chooses an arbitrary header H, message M, keys (K^e, K^m), and randomness R, and computes $R \,\|\, C \leftarrow \mathsf{enc}_{K^e}(M\ ;\ R)$ and then $T \leftarrow F_{K^m}(H \,\|\, R \,\|\, C)$. It then chooses another key $\widetilde{K}^e \neq K^e$, and computes $\widetilde{M} \leftarrow \mathsf{dec}_{\widetilde{K}^e}(R \,\|\, C)$. Finally, it outputs $(H, (R, (K^e, K^m))), (H, (R, (\widetilde{K}^e, K^m))), T)$. It is easy to check that this adversary will win the r-BIND game with probability close to one, assuming SE is such that decrypting the same ciphertext under different keys yields distinct plaintexts with overwhelming probability.

```
Enc(K, H, M):
    $K^e, K^m \leftarrow K$
    $IV \leftarrow\$ \{0,1\}^n$
    $T \leftarrow \mathrm{RO}_{K^m}(H \parallel M)$
    $C \leftarrow \mathrm{CBC}_{K^e}(\mathrm{Pad}_n(M \parallel T)\,;\, IV)$
    $\ell \leftarrow \mathrm{Pad}_n(M \parallel T)/n$
    $C' \parallel C_{\ell-2} \parallel C_{\ell-1} \parallel C_\ell \leftarrow C$
    Return $(C', IV \| C_{\ell-2} \| C_{\ell-1} \| C_\ell)$
```

```
Dec(K, H, C_1, C_2):
    $K^e, K^m \leftarrow K$
    $IV \parallel C_{\ell-2} \parallel C_{\ell-1} \parallel C_\ell \leftarrow C_2$
    $C_f \leftarrow C_{\ell-2} \parallel C_{\ell-1} \parallel C_\ell$
    $M \| T \leftarrow \mathrm{CBC}^{-1}_{K^e}(C_1 \| C_f\,;\, IV)$
    $T' \leftarrow \mathrm{RO}_{K^m}(H \parallel M)$
    If $T \neq T'$ then Return $\perp$
    Return $(M, (K^e, K^m))$
```

```
Ver(H, M, K_f, C_2):
    $K^e, K^m \leftarrow K_f$
    $IV \parallel C'_{\ell-2} \parallel C'_{\ell-1} \parallel C'_\ell \leftarrow C_2$
    $T \leftarrow \mathrm{RO}_{K^m}(H \parallel M)$
    $P \leftarrow \mathrm{Pad}_n(M \parallel T)$
    $C \leftarrow \mathrm{CBC}_{K^e}(P\,;\, IV)$
    $C' \parallel C''_{\ell-2} \parallel C''_{\ell-1} \parallel C''_\ell \leftarrow C$
    Return $\bigwedge_{i=\ell-2}^{\ell} (C''_i = C'_i)$
```

Fig. 7. Committing authenticated encryption based on MtE composition of CBC mode and a MAC modeled as a random oracle. The length ℓ is defined to be $\mathrm{Pad}_n(M \parallel T)/n$. The function Pad is the standard PKCS#7 padding used in TLS. The notation $\mathrm{CBC}_K(\cdot\,;\, IV)$ and $\mathrm{CBC}^{-1}_K(\cdot\,;\, IV)$ means CBC mode encryption and decryption with key K and initialization vector IV.

5.3 MAC-then-Encrypt

The MAC-then-encrypt mode generically composes a MAC and an encryption scheme by first computing the MAC of the header and message, then appending the MAC to the message and encrypting them both. The pseudcode in Figure 7 uses for concreteness CBC mode encryption and we refer to this committing AEAD scheme as MtE. We will also assume the MAC is suitable to be modeled as a keyed random oracle; HMAC-SHA256 is one such [28]. CBC with HMAC in an MtE mode is a common cipher suite for modern TLS connections, which motivated these choices. Prior work has investigated the security of MtE in the sense of CTXT [42, 51] and its ROR security is inherited directly from the encryption mode. Below we will assume that the block size of n bits for the cipher underlying CBC mode, and that our MACs have output length $2n$ bits.

Unlike with Encrypt-then-MAC, we are able to prove the two-key version of MtE secure in the sense of receiver binding. The binding security of MtE in the case where keys are derived via a KDF follows as a corollary, though we believe better bounds can be achieved in this case.

A sketch of an argument that MtE is binding (in the traditional sense where the entire ciphertext is the commitment) appeared in [58]. Their approach, which only relied on modeling the MAC as a RO and made no assumptions about CBC mode, led to a rather loose bound. We instead additionally model the cipher underlying CBC as ideal. This results in a simpler and tighter proof. Our proof, given in the full version of the paper, can also be readily adapted to when CTR mode is used instead of CBC.

Theorem 4. *Let* MtE *be the scheme defined above using a random oracle and an ideal cipher within CBC mode. For any* r-BIND$_{\mathsf{MtE}}$ *adversary* $\mathcal{A}$ *making at most* q_i *queries to its ideal cipher and* q_r *queries to its random oracle, it holds that* $\mathbf{Adv}^{\text{r-bind}}_{\mathsf{MtE}}(\mathcal{A}) < q_i q_r / 2^{2n}$.

The s-BIND advantage against compactly-committing MtE is zero, since the commitment along with the output of a successful call to Dec uniquely defines the inputs to Ver. Thus, no other ciphertext can be computed in Ver other than the one previously decrypted in Dec, because the inputs to Ver are fixed by Dec.

5.4 Some Non-binding AEAD schemes

In this section we will briefly detail attacks which break the receiver binding security of some deployed AEAD schemes. In particular, typical schemes that use MACs which are not collision resistant, such as Carter-Wegman MACs, do not suffice. For completeness we spell out an example of breaking the receiver binding of GCM [46], an encrypt-then-MAC style construction that uses a Carter-Wegman MAC.

A slight simplification of the GCM MAC is the function F shown in Figure 8 applied to a ciphertext. (We ignore associated data for simplicity.) It uses a key K for a block cipher E with block size n, as well as a nonce N. An initial point $P_0 \leftarrow E_K(0^n)$ and a pad $R \leftarrow E_K(N)$ are computed. GCM uses an ϵ-almost XOR universal (ϵ-AXU) [57] hash function computed by considering a ciphertext of m encrypted message blocks an m-degree polynomial defined over a finite field $\mathbb{F}$. The field is a particular representation of GF(2^{128}). This polynomial is evaluated at the encryption point P_0 and the result is XOR'd with the pad R. The GCM AEAD scheme encrypts the message using CTR mode encryption using E_K and a random 96-bit IV concatenated with a 32-bit counter initially set at one, and then MACs the resulting ciphertext $C = C_1, \ldots, C_m$ to generate a tag $T = F(K, IV \parallel 0^{32}, C_1, \ldots, C_m)$.

A straightforward way to consider GCM as a compactly committing AEAD is to have encryption output as the commitment portion C_2 the tag T, and the rest of the ciphertext as the first portion C_1. Decryption works as usual for GCM, but additionally outputs (IV, K) as the opening. Verification works by recomputing encryption and checking that the resulting tag matches the commitment value C_2. We denote this scheme simply by $\mathsf{GCM} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Ver})$ below.

```
$F(K, N, (C_1, \ldots, C_m))$:
$P_0 \leftarrow E_K(0^n)$
$R \leftarrow E_K(N)$
$S \leftarrow \sum_{i=1}^{m} C_i P_0^{m-i}$
$T \leftarrow R \oplus S$
Return $T$
```

Fig. 8. A simplified description of the CW MAC used in GCM.

We now give an r-BIND$_{\mathsf{GCM}}$ adversary $\mathcal{A}$. We ignore associated data for simplicity. To win, $\mathcal{A}$ must output $((M, (IV, K), (M', (IV', K'), T)$ so that $\mathsf{Ver}(M, (IV, K), T) = \mathsf{Ver}(M', (IV', K'), T) = 1$. We will build an $\mathcal{A}$ that chooses messages such that $|M| = |M'|$. The adversary $\mathcal{A}$ will start by choosing a ciphertext $C_1, \ldots, C_m$ such that

$$F(K, IV, C_1, \ldots, C_m) = F(K', IV', C_1, \ldots, C_m) \tag{1}$$

and letting M (resp. M') be the CTR-mode decryption of $C_1, \ldots, C_m$ under IV, K (resp. IV', K'). Choosing the ciphertext such that condition 1 holds is straightforward, as plugging in for the definition of F and rearranging, the adversary must solve the equation

$$\left[\sum_{i=1}^{m} C_i(P^{m-i} + (P')^{m-i})\right] + (E_K(N) + E_{K'}(N')) = 0$$

```
CtE1-Enc(K, H, M)
    $(K_f, C_2) \leftarrow\!\!\$\ \mathsf{Com}(H \| M)$
    $C_1 \leftarrow\!\!\$\ \mathsf{enc}_K(C_2, M \| K_f)$
    Return $(C_1, C_2)$
```

```
CtE2-Enc(K, H, M)
    $(K_f, C_2) \leftarrow\!\!\$\ \mathsf{Com}(H \| M)$
    $C_1 \leftarrow\!\!\$\ \mathsf{enc}_K(H, M \| K_f)$
    Return $(C_1, C_2)$
```

```
CtE1-Dec(K, H, C_1, C_2)
    $(M \| K_f) \leftarrow \mathsf{dec}_K(C_2, C_1)$
    If $M = \perp$ then Return $\perp$
    $b \leftarrow \mathsf{VerC}(K_f, C_2, H \| M)$
    If $b = 0$ then
        Return $\perp$
    Return $(M, K_f)$
```

```
CtE2-Dec(K, H, C_1, C_2)
    $(M \| K_f) \leftarrow \mathsf{dec}_K(H, C_1)$
    If $M = \perp$ then Return $\perp$
    $b \leftarrow \mathsf{VerC}(K_f, C_2, H \| M)$
    If $b = 0$ then
        Return $\perp$
    Return $(M, K_f)$
```

Fig. 9. Algorithms for two Commit-then-Encrypt variants. Facebook's scheme uses CtE2 with an HMAC-based commitment. CtE1-Ver and CtE2-Ver both just output $\mathsf{VerC}(H, M, K_f, C_2)$.

where $P \leftarrow E_K(0^n)$ and $P' \leftarrow E_{K'}(0^n)$. For example, pick arbitrary $C_1, \ldots, C_{m-1}$ and solve for the C_m that satisfies the equation.

This attack works even if associated data is used, or if the whole ciphertext is used as the commitment. A very similar attack works on ChaCha20/Poly1305 [15]; a small tweak is required to handle the fact that not every member of $\mathbb{F}_{2^{130}-5}$ is a valid ciphertext block.

6 Composing Commitment and AEAD

In the last section we saw that existing AEAD schemes already realize (compactly) committing AEAD in some cases. These schemes, however, only realize single-opening security as the opening includes the secret key. We now turn to schemes that achieve multi-opening committing AEAD, and focus specifically on schemes that generically compose AEAD with a commitment scheme.

Commit-then-Encrypt. We start with a simple general construction, what we call the Commit-then-Encrypt scheme.[5] It combines a commitment scheme $\mathsf{CS} = (\mathsf{Com}, \mathsf{VerC})$ with an AEAD scheme $\mathsf{SE} = (\mathsf{Kg}, \mathsf{enc}, \mathsf{dec})$. Formally the scheme $\mathrm{CtE1}[\mathsf{CS}, \mathsf{SE}] = (\mathsf{Kg}, \mathrm{CtE1\text{-}Enc}, \mathrm{CtE1\text{-}Dec}, \mathrm{CtE1\text{-}Ver})$ works as shown in Figure 9.

The CtE1 scheme produces a commitment value to the message and associated data H, and then encrypts the message along with the opening of the commitment. It uses as associated data during encryption the commitment value, but not H. This nevertheless binds the underlying AEAD ciphertext to H as well as C_2 — as we will show tampering with either will be detected and rejected during decryption. One could additionally include H in the associated data for enc, but this would be less efficient. Should a protocol want H to not be in the commitment scope, one can instead include H only as associated data within enc and omit it from the commitment.

The proofs of the next two theorems will appear in the full version.

[5] This name was also used in [36], but the scheme is distinct. See Section 9.

Theorem 5 (CtE1 **confidentiality).** *Let* CtE1 = CtE1[CS, SE]. *Let $\mathcal{A}$ be an* MO-ROR$_{\text{CtE1}}$ *adversary making at most q queries to its oracles. Then we give adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{C}$ such that*

$$\mathbf{Adv}^{\text{mo-ror}}_{\text{CtE1}}(\mathcal{A}) \leq \mathbf{Adv}^{\text{ror}}_{\mathsf{SE}}(\mathcal{B}_1) + \mathbf{Adv}^{\text{ror}}_{\mathsf{SE}}(\mathcal{B}_2) + \mathbf{Adv}^{\text{cs-ror}}_{\mathsf{CS}}(\mathcal{C}) \ .$$

The adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, and $\mathcal{C}$ all make the same number of queries as $\mathcal{A}$ and all run in time that of $\mathcal{A}$ plus at most $\mathcal{O}(q)$ overhead.

Theorem 6 (CtE1 **ciphertext integrity).** *Let* CtE1 = CtE1[CS, SE]. *Let $\mathcal{A}$ be an* MO-CTXT$_{\text{CtE1}}$ *adversary making at most q queries to its oracles. Then we give adversaries $\mathcal{B}$, $\mathcal{C}$ such that*

$$\mathbf{Adv}^{\text{mo-ctxt}}_{\text{CtE1}}(\mathcal{A}) \leq \mathbf{Adv}^{\text{ctxt}}_{\mathsf{SE}}(\mathcal{B}) + \mathbf{Adv}^{\text{v-bind}}_{\mathsf{CS}}(\mathcal{C}) \ .$$

Adversary $\mathcal{B}$ makes the same number of queries as $\mathcal{A}$ and runs in time that of $\mathcal{A}$ plus at most $\mathcal{O}(q)$ overhead. Adversary $\mathcal{C}$ runs in time that of $\mathcal{A}$.

The receiver binding security of CtE1 is trivially implied by the security of the underlying commitment scheme, as captured by the next theorem.

Theorem 7 (CtE1 **receiver binding).** *Let* CtE1 = CtE1[CS, SE]. *Let $\mathcal{A}$ be an* r-BIND$_{\text{CtE1}}$ *adversary. Then* $\mathbf{Adv}^{\text{r-bind}}_{\text{CtE1}}(\mathcal{A}) = \mathbf{Adv}^{\text{v-bind}}_{\mathsf{CS}}(\mathcal{A})$.

We conclude the section by noting CtE1 meets s-BIND security, since it runs Ver during decryption.

Facebook's scheme. The Facebook franking scheme (Section 3) is almost, but not quite, an instantiation of CtE1 using HMAC as the commitment scheme CS. One difference is that their franking scheme does not bind C_2 to C_1 by including C_2 in the associated data during encryption. The other difference is that the Facebook scheme builds a commitment from HMAC by first generating a random secret key, then using it to evaluate HMAC on the concatenation of the message and the key itself (see Figure 2 for a diagram). Assuming HMAC remains a collision-resistant PRF when evaluated on its own key, we can prove Facebook's non-standard construction is a secure commitment (see Theorem 1).

To analyze Facebook's scheme, then, we introduce the scheme CtE2[SE, CS] = (Kg, CtE2-Enc, CtE2-Dec, CtE2-Ver) that works as shown in Figure 9. Note that Facebook does not discuss how to handle associated data, and so their scheme is CtE2 using CS instantiated with HMAC and requiring $H = \varepsilon$.

There are two benefits to the approach of CtE1: (1) proving ciphertext integrity does not require any special properties of the commitment scheme, and (2) it is more efficient because associated data is cryptographically processed once, rather than twice. We therefore advocate CtE1, but analyze CtE2 here since it is already in use.

CtE2 is *not* secure assuming just that CS is hiding and binding. The reason is that such commitments can be malleable and this allows easy violation of ciphertext integrity. Specifically, consider a commitment scheme CSBad =

$(\mathsf{ComBad}, \mathsf{VerBad})$ built using a standard commitment scheme $\mathsf{CS} = (\mathsf{Com}, \mathsf{VerC})$. Algorithm $\mathsf{ComBad}(M)$ runs $(K_c, C) \leftarrow\!\!\$\ \mathsf{Com}(M)$ and then outputs $(K_c, C \,\|\, 1)$. Algorithm $\mathsf{VerBad}(M, K_f, C \,\|\, b)$ runs $\mathsf{VerC}(M, K_f, C)$ and outputs the result. An easy reduction shows that CSBad is both hiding and binding, assuming CS is too. But it's clear that $\mathrm{CtE2}[\mathsf{SE}, \mathsf{CSBad}]$ does *not* enjoy ciphertext integrity. The adversary simply obtains one ciphertext, flips the last bit, and submits to the challenge decryption oracle to win.

This shows that standard commitments with hiding and binding properties are insufficient to instantiate CtE2. But if a scheme CS has unique commitments, then we can in fact show security of CtE2. A scheme has *unique commitments* if for any pair $(K_c, M) \in \mathcal{K}_f \times \mathcal{M}$ it holds that there is a single commitment value $C \in \mathcal{C}$ for which $\mathsf{Ver}(K_c, C, M) = 1$. All hash-based CS schemes, including the HMAC one used by Facebook's franking scheme, have unique commitments. If one wanted to use a scheme that does not have unique commitments, then one would need the commitment to satisfy a form of non-malleability [29].

The following sequence of theorems captures the security of CtE2 assuming a unique commitment scheme. Proofs appear in the full version.

Theorem 8 **(**CtE2 **confidentiality).** *Let* $\mathrm{CtE2} = \mathrm{CtE2}[\mathsf{CS}, \mathsf{SE}]$. *Let $\mathcal{A}$ be an* $\text{MO-ROR}_{\mathrm{CtE2}}$ *adversary making at most q oracle queries. Then we give adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{C}$ such that*

$$\mathbf{Adv}^{\text{mo-ror}}_{\mathrm{CtE2}[\mathsf{CS},\mathsf{SE}]}(\mathcal{A}) \le \mathbf{Adv}^{\text{ror}}_{\mathsf{SE}}(\mathcal{B}_1) + \mathbf{Adv}^{\text{ror}}_{\mathsf{SE}}(\mathcal{B}_2) + \mathbf{Adv}^{\text{cs-ror}}_{\mathsf{CS}}(\mathcal{C})$$

Adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, and $\mathcal{C}$ all run in time that of $\mathcal{A}$ plus at most $\mathcal{O}(q)$ overhead and make at most q queries.

Theorem 9 **(**CtE2 **ciphertext integrity).** *Let* $\mathrm{CtE2} = \mathrm{CtE2}[\mathsf{CS}, \mathsf{SE}]$ *and assume* CS *has unique commitments. Let $\mathcal{A}$ be an* $\text{MO-CTXT}_{\mathrm{CtE2}}$ *adversary making at most q queries. Then we give adversaries $\mathcal{B}$, $\mathcal{C}$ such that*

$$\mathbf{Adv}^{\text{mo-ctxt}}_{\mathrm{CtE2}[\mathsf{CS},\mathsf{SE}]}(\mathcal{A}) \le \mathbf{Adv}^{\text{ctxt}}_{\mathsf{SE}}(\mathcal{B}) + \mathbf{Adv}^{\text{v-bind}}_{\mathsf{CS}}(\mathcal{C}) \ .$$

Adversaries $\mathcal{B}$ and $\mathcal{C}$ both run in time that of as $\mathcal{A}$ plus at most $\mathcal{O}(q)$ overhead. Adversary $\mathcal{B}$ makes at most q queries to its oracles.

Theorem 10 **(**CtE2 **receiver binding).** *Let* $\mathrm{CtE2} = \mathrm{CtE2}[\mathsf{CS}, \mathsf{SE}]$. *Let $\mathcal{A}$ be an* $\text{r-BIND}_{\mathrm{CtE2}}$ *adversary. Then we give an adversary $\mathcal{B}$ such that*

$$\mathbf{Adv}^{\text{r-bind}}_{\mathrm{CtE1}[\mathsf{CS},\mathsf{SE}]}(\mathcal{A}) \le \mathbf{Adv}^{\text{v-bind}}_{\mathsf{CS}}(\mathcal{B}) \ .$$

Adversary $\mathcal{B}$ runs time that of $\mathcal{A}$.

Finally, note that CtE2 achieves s-BIND security because it verifies the commitment during decryption.

7 Nonce-based Committing AEAD and the CEP Construction

The committing AEAD schemes thus far have all been randomized. Cryptographers have advocated that modern AEAD schemes, however, be designed to be nonce-based. Here one replaces internal randomness during encryption with an input, called the nonce. Security should hold as long as the nonce never repeats throughout the course of encrypting messages with a particular key.

We formalize nonce-based committing AEAD and provide a construction of it that additionally achieves a number of valuable properties. It will achieve a multiple-opening security notion suitably modified to the nonce-based setting. It is faster than the other multiple-opening schemes, requiring only two cryptographic passes during encryption and decryption, and a single one during verification. It also reduces ciphertext stretch compared to the schemes of Section 6, since the opening will be recomputed in the course of decryption and so does not need to be sent in the encryption.

Nonce-based committing AEAD. A nonce-based CE scheme is a tuple of algorithms $\mathrm{nCE} = (\mathrm{Kg}, \mathrm{Enc}, \mathrm{Dec}, \mathrm{Ver})$. We define it exactly like CE schemes (Section 4) except for the following differences. In addition to the other sets, we associate to any nCE scheme a nonce space $\mathcal{N} \subseteq \Sigma^*$. Encryption and decryption are now defined as follows:

- *Encryption*: Encryption Enc is deterministic and takes as input a tuple $(K, N, H, M) \in (\Sigma^*)^4$ and outputs a pair $(C_1, C_2) \in \mathcal{C} \times \mathcal{T}$ or a distinguished error symbol $\perp$. We require that for any $(K, N, H, M) \in \mathcal{K} \times \mathcal{N} \times \mathcal{H} \times \mathcal{M}$ it is the case that $\mathrm{Enc}(K, N, H, M) \neq \perp$.
- *Decryption*: Decryption Dec is deterministic. It takes as input a quintuple $(K, N, H, C_1, C_2) \in (\Sigma^*)^5$ and outputs a message, opening value pair $(M, K_f) \in \mathcal{M} \times \mathcal{K}_f$ or $\perp$.

Key generation and verification are unchanged relative to randomized CE schemes. As for randomized schemes, we assume that the length of ciphertexts are dictated only by the lengths of the header and message. We will often write $\mathrm{Enc}_K^N(H, M)$ for $\mathrm{Enc}(K, N, H, M)$ and $\mathrm{Dec}_K^N(H, C_1, C_2)$ for $\mathrm{Dec}(K, N, H, C_1, C_2)$.

Nonce-based security. We adapt the confidentiality and integrity security notions from Section 4 to the nonce-based setting. Let game $\text{MO-nREAL}_{\mathrm{nCE}}^{\mathcal{A}}$ be the same as the game $\text{MO-RAND}_{\mathrm{nCE}}^{\mathcal{A}}$ (Figure 3), except that all oracles take an additional input N, Enc and Dec executions use that value N as the nonce, the sets $\mathcal{Y}_1, \mathcal{Y}_2$ are instead updated with (N, H, C_1, C_2), and the decryption oracle checks if $(N, H, C_1, C_2) \in \mathcal{Y}_1$. Similarly let game $\text{MO-nRAND}_{\mathrm{nCE}}^{\mathcal{A}}$ be the same as $\text{MO-RAND}_{\mathrm{nCE}}^{\mathcal{A}}$ (Figure 3), except that all oracles take an additional input N, and Enc and Dec use that value N as the nonce, and $\mathcal{Y}_2$ is updated with (N, H, C_1, C_2). For a scheme nCE, we measure the nonce-based multiple-openings real-or-random $\text{MO-nROR}_{\mathrm{nCE}}$ advantage of an adversary $\mathcal{A}$ by

$$\mathbf{Adv}_{\mathrm{nCE}}^{\text{mo-nror}}(\mathcal{A}) = \left|\Pr\left[\,\text{MO-nREAL}_{\mathrm{nCE}}^{\mathcal{A}} \Rightarrow 1\,\right] - \Pr\left[\,\text{MO-nRAND}_{\mathrm{nCE}}^{\mathcal{A}} \Rightarrow 1\,\right]\right| .$$

An adversary is *nonce-respecting* if its queries never repeat the same N across a pair of encryption queries (two queries to **Enc**, two to **ChalEnc**, or one to each). We will assume nonce-respecting $\text{MO-nRAND}_{\text{nCE}}$ adversaries.

Let $\text{MO-nCTXT}_{\text{nCE}}^{\mathcal{A}}$ be the same as the game $\text{MO-CTXT}_{\text{nCE}}^{\mathcal{A}}$ (Figure 3), except that all oracles take an additional input N, **Enc** and **Dec** executions use that value N as the nonce, and the set $\mathcal{Y}$ is instead updated with (N, H, C_1, C_2). For a scheme nCE, we measure the nonce-based multiple-openings real-or-random $\text{MO-nCTXT}_{\text{nCE}}$ advantage of an adversary $\mathcal{A}$ by

$$\mathbf{Adv}_{\text{nCE}}^{\text{mo-nctxt}}(\mathcal{A}) = \Pr\left[\, \text{MO-nCTXT}_{\text{nCE}}^{\mathcal{A}} \Rightarrow 1 \,\right] \ .$$

As with randomized committing AEAD, we can provide single-opening versions of the above definitions, and can give an all-in-one version of nonce-based MO and SO security. We omit the details for the sake of brevity.

The sender binding notion s-BIND for nonce-based schemes is the same as for randomized schemes except that the adversary also outputs a nonce N, which is used with Dec. Because verification is unchanged, receiver binding security is formalized exactly the same for randomized and nonce-based committing AEAD.

The Committing Encrypt-and-PRF scheme. One can analyze some traditional nonce-based AEAD schemes to show they are compactly committing. As one example, it is easy to see that the EtE construction (Section 5.1) works just as well with non-repeating nonces, but with only single-opening security. The other schemes in Section 5 do not, but can be easily modified to by replacing IV with $E_K(N)$. Here we focus on a new scheme that will have better overall performance and security than previous ones. Unlike the legacy schemes studied in Section 5 it will be provably secure for multiple openings. At the same time, it will be more efficient than the schemes in Section 6.

The scheme $\text{CEP}[G[K], F, F^{\text{cr}}] = (\text{Kg}, \text{CEP-Enc}, \text{CEP-Dec}, \text{CEP-Ver})$ is in the style of an Encrypt-and-PRF construction. It uses an underlying stream cipher $G[E]$ built from a block cipher $E\colon \{0,1\}^k \times \{0,1\}^n \times \{0,1\}^n$ and functions $F, F^{\text{cr}}\colon \{0,1\}^n \times \{0,1\}^t \to \{0,1\}^t$. The key space is $\mathcal{K} = \{0,1\}^k$ and key generation simply outputs a random draw from it. Encryption starts by using the nonce with the key K to derive one-time keys for the keyed cryptographic hash F^{cr} and a PRF F, as well as to generate an encryption pad to XOR with the message. We use a block cipher E in CTR mode to generate these values. Finally it computes a binding value for H, M and applies F to that commitment value to generate a tag. Detailed pseudocode is given in Figure 10.

We will need F^{cr} to both be CR (for binding) as well as secure as a one-time PRF (for confidentiality). This rules out some otherwise desirable choices such as CMAC [41], PMAC [53] and Carter-Wegman-style PRFs such as Poly1305 [16] and UMAC [17]. These PRFs are some of the fastest available, but would make CEP vulnerable to binding attacks. (See also the discussion in Section 5.4.)

The most obvious choice is HMAC, for which formal analyses support it being a secure PRF for a key secret [2, 3] and CR for adversarially chosen keys of the same length (assuming the underlying hash function is CR). Other multi-property hash functions [10] would also suffice.

```
CEP-Enc$^N_K(H, M)$:
$IV \leftarrow E_K(N)$
$m \leftarrow \lceil |M|/n \rceil$
For $i = 0$ to $m + 1$ do
    $P_i \leftarrow E_K(IV + i)$
$C_1 \leftarrow (P_2 \parallel \cdots \parallel P_{m+1}) \oplus M$
$C_2 \leftarrow F^{\mathrm{cr}}_{P_0}(H \parallel M)$
$T \leftarrow F_{P_1}(C_2)$
Return $(C_1 \parallel T, C_2)$
```

```
CEP-Dec$^N_K(H, C_1 \parallel T, C_2)$:
$m \leftarrow \lceil |C_1|/n \rceil$
For $i = 0$ to $m + 1$ do
    $P_i \leftarrow E_K(IV + i)$
$M \parallel \leftarrow (P_2 \parallel \cdots \parallel P_{m+1}) \oplus C_1$
$C'_2 \leftarrow F^{\mathrm{cr}}_{P_0}(H \parallel M)$
$T' \leftarrow F_{P_1}(C'_2)$
If $T \neq T' \vee C'_2 \neq C_2$ then
    Return $\perp$
Return $(M, P_0)$
```

```
CEP-Ver$(H, M, K_f, C_2)$:
$C'_2 \leftarrow F^{\mathrm{cr}}_{K_f}(H \parallel M)$
If $C'_2 \neq C_2$ then Return 0
Return 1
```

Fig. 10. A nonce-based committing AEAD.

The reason we use E_K both for CTR mode and for key derivation is speed. This ensures that we need ever only use a single key with E; in some environments rekeying can be almost as expensive as another invocation of E. In fact we are simply using E_K to build a stream cipher, and any nonce-based secure stream cipher would do, e.g., ChaCha-20 [15].

One might wonder why have a tag T as well as the commitment value C_2. The reason is that to achieve multi-opening security, we must disclose the key used with F^{cr}, rendering the unforgeability of C_2 values moot. If one instead omitted T and only checked $C'_2 = C_2$ to attempt to achieve unforgeability, then there exists a straightforward MO-nCTXT attack that obtains a ciphertext for a nonce N, queries it to **Dec** to get the key for F^{cr}, and then uses that to forge a new ciphertext to be submitted to **ChalDec**. The application of F under a distinct key provides ciphertext integrity even after an adversary obtains openings (keys for F^{cr}). Similarly, dropping the check during decryption that $C'_2 = C_2$ also leads to an attack, but this time on sender binding.

Comparisons. Before getting into the formal security analysis in the next section, we first compare CEP to the generic composition constructions that also achieve multiple-opening security. The first benefit over other schemes is that it is nonce-based, making it suitable for stateful as well as randomized settings (see also Rogaway's discussion of the benefits of nonce-based encryption [54]).

The second is that ciphertext expansion is reduced by a security parameter number of bits compared to the generic composition constructions, because in CEP we do not need to transport an explicit opening — the recipient recomputes it pseudorandomly from the secret key. Consequently, CEP ciphertexts are shorter than Facebook's by 256 bits.

The third is that encryption and decryption both save an entire cryptographic pass over the associated data and message. For Facebook's chosen algorithms (HMAC for the commitment, plus Encrypt-then-MAC using AES-CBC and HMAC), this means that CEP offers more than a 50% speed-up for both algorithms.[6] While in some messaging settings encryption and decryption may not

[6] HMAC is slower than AES. If AES-NI is available, then the speed-up will be even larger, since the HMAC passes will be the performance bottleneck.

be particularly performance-sensitive operations, any cost savings is desirable. In other contexts, such as if one starts using committing encryption on larger files (images, videos) sent over messaging applications or if one wants abuse reporting for streaming communications, performance will be very important.

CEP achieves the stronger multiple-opening security goal, setting it apart from the legacy committing AEAD schemes from Section 5. At the same time, CEP has equivalent or better performance than those schemes. With respect to EtM and MtE, verification is reduced from two cryptographic passes to one.

8 Analysis of CEP

Useful abstractions. We will introduce some intermediate abstractions of the underlying primitives. First, a nonce-based stream cipher G takes as input a key K, a nonce N, and an output length ℓ. It outputs a string of length ℓ bits. The second abstraction is of the implicit MAC used within CEP. It is the composition $F \circ F^{\mathrm{cr}}(P_1, P_0, H \| M) = F_{P_1}(F^{\mathrm{cr}}_{P_0}(H \| M))$ for random keys P_0, P_1 and any strings H, M. The output is a t-bit string. We defer a discussion of the security properties required from these abstractions to the full version of this work. There, we define a nonce-based pseudorandom generator (PRG) security notion that mandates attackers cannot distinguish between G's output and random bit strings, as well as a multi-user unforgeability notion MU-UF-CMA$_{F \circ F^{\mathrm{cr}}}$ that captures the unforgeability of $F \circ F^{\mathrm{cr}}$ when adversaries can attack it under multiple keys.

Security of CEP. We are now in position to formally analyze the confidentiality, ciphertext integrity, and binding of CEP. We give theorems for each in turn, with proofs deferred to the full version of the paper.

Theorem 11 (CEP **confidentiality**). *Let* CEP = CEP$[G, F]$. *Let* $\mathcal{A}$ *be an* MO-nROR$_{\mathrm{CEP}}$ *adversary making at most q queries and whose queried messages total at most σ bits. Then we give adversaries $\mathcal{B}, \mathcal{C}, \mathcal{D}$ such that*

$$\mathbf{Adv}^{\mathrm{mo\text{-}nror}}_{\mathrm{CEP}}(\mathcal{A}) \leq 2 \cdot \mathbf{Adv}^{\mathrm{prg}}_{G}(\mathcal{B}) + 2 \cdot \mathbf{Adv}^{\mathrm{prf}}_{F}(\mathcal{C}) + \cdot \mathbf{Adv}^{\mathrm{prf}}_{F^{\mathrm{cr}}}(\mathcal{D})$$

Adversary $\mathcal{B}$ makes at most q queries to its oracle, the sum of its total outputs requested is σ bits. Adversary $\mathcal{C}$ makes at most q queries to its oracle, and never repeats a key identifier. Adversary $\mathcal{D}$ make at most q queries to its oracle and never repeats a key identifier. All adversaries run in time at most that of $\mathcal{A}$ plus an overhead of at most $\mathcal{O}(q)$.

Theorem 12 (CEP **ciphertext integrity**). *Let* CEP = CEP$[G, F]$. *Let* $\mathcal{A}$ *be an* MO-nCTXT$_{\mathrm{CEP}}$ *adversary making at most q queries with query inputs totalling at most σn bits. Let F^2 be the tagging scheme described earlier. Then we give adversaries $\mathcal{B}, \mathcal{C}$ such that*

$$\mathbf{Adv}^{\mathrm{mo\text{-}nctxt}}_{\mathrm{CEP}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathrm{prg}}_{G}(\mathcal{B}) + \mathbf{Adv}^{\mathrm{mu\text{-}uf\text{-}cma}}_{F^2}(\mathcal{C}) \; .$$

Adversary $\mathcal{B}$ runs in time that of $\mathcal{A}$ plus at most $\mathcal{O}(q)$ overhead and makes q queries totaling at most σn bits. Adversary $\mathcal{C}$ makes at most q queries and runs in time that of $\mathcal{A}$ plus at most $\mathcal{O}(q)$ overhead.

Finally we turn to binding. Recall that any scheme that effectively runs commitment verification during decryption achieves sender binding. The check that $C_2' = C_2$ during decryption accomplishes this, and so the scheme is perfectly sender binding. For receiver binding, a simple reduction gives the following theorem showing that the CR of F^{cr} implies binding of CEP.

Theorem 13 (CEP **receiver binding).** *Let* $\mathrm{CEP} = \mathrm{CEP}[G, F]$. *Let* $\mathcal{A}$ *be any* $\text{r-BIND}_{\mathrm{CEP}}$ *adversary. Then we give an adversary* $\mathcal{B}$ *such that* $\mathbf{Adv}^{\text{r-bind}}_{\mathrm{CEP}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathrm{cr}}_{F^{\mathrm{cr}}}(\mathcal{B})$ *and* $\mathcal{B}$ *runs in time that of* $\mathcal{A}$.

9 Related Work

The primary viewpoint in the literature has been that committing encryption is undesirable either because one wants deniability [20, 22, 50] or due to the theoretical challenges associated with proving encryption confidentiality in the face of adaptive compromises [23]. Thus while *non*-committing encryption has received significant attention (q.v., [22–25, 27, 34, 35, 40, 43, 49, 50, 62–65]), there is a dearth of literature on building purposefully committing encryption.

We are aware of only one previous work on building committing encryption schemes, due to Gertner and Herzberg [36]. They give definitions that are insufficient for the message franking setting (in particular they do not capture server binding or multiple opening security). They do not analyze AE schemes, and focus only on building asymmetric primitives.

Our receiver binding security property is related to the concept of robust encryption, introduced by Abdalla et al. [1]. They give two security notions for public-key encryption (PKE). The stronger, called strong robustness, asks that an adversarially-chosen ciphertext should only correctly decrypt under at most one legitimate secret key. Mohassel [48] showed efficient ways of adapting existing PKE schemes to be robust. Farshim et al. [32] subsequently pointed out that some applications require robustness to adversarially generated secret keys, and introduced a notion called complete robustness. In a later work, Farshim, Orlandi, and Rosie [33] adapt these robustness definitions to the setting of authenticated encryption, message authentication codes (MACs), and pseudorandom functions (PRFs). They show that in this context, the simpler full robustness notion of [32] is the strongest of those considered.

These prior notions, in particular the full robustness for AE notion from [33], do not suffice for formalizing binding for AEAD. First, it does not capture sender binding. Second, for receiver binding, it turns out that the most straightforward adaptation of full robustness to handle associated data fails to imply receiver binding. We defer a more detailed explanation to the full version of this work.

Abdalla et al. [1] propose a generic composition of a commitment scheme and PKE scheme to achieve robustness and Farshim et al. [33] show a variant of this for the symmetric encryption setting. The latter construction commits to the key, not the message, and could not be used to achieve the multiple opening security targeted by our generic composition constructions.

Selective-opening security allows an adversary to adaptively choose to corrupt some senders that sent (correlated) encrypted messages [8] or to compromise the keys of a subset of receivers [38]. Bellare et al. [8] gave the first constructions of schemes secure against selective-opening attacks for sender corruptions. Non-committing encryption can be used to realize security for receiver corruptions. Our definitions do not model selective-opening attacks, and as mentioned in the introduction, assessing the viability of committing AEAD in selective-opening settings is an interesting open problem.

Acknowledgments

The authors would like to thank the anonymous reviewers of Crypto 2017 for their thoughtful comments, as well as Mihir Bellare for discussions about robust encryption and its relation to binding. This work was funded in part by NSF grant CNS-1330308.

References

1. Michel Abdalla, Mihir Bellare, and Gregory Neven. Robust encryption. In *Theory of Cryptography Conference*, 2010.
2. Mihir Bellare. New proofs for nmac and hmac: Security without collision-resistance. In *CRYPTO*, 2006.
3. Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. In *CRYPTO*, 1996.
4. Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Pseudorandom functions revisited: The cascade construction and its concrete security. In *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*, pages 514–523. IEEE, 1996.
5. Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Hmac: Keyed-hashing for message authentication. *Internet Request for Comment RFC*, 2104, 1997.
6. Mihir Bellare, Anand Desai, Eron Jokipii, and Phillip Rogaway. A concrete security treatment of symmetric encryption. In *Foundations of Computer Science (FOCS)*, 1997.
7. Mihir Bellare, Rafael Dowsley, Brent Waters, and Scott Yilek. Standard security does not imply security against selective-opening. In *EUROCRYPT*, 2012.
8. Mihir Bellare, Dennis Hofheinz, and Scott Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In *EUROCRYPT*, 2009.
9. Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In *ASIACRYPT*, 2000.
10. Mihir Bellare and Thomas Ristenpart. Multi-property-preserving hash domain extension and the EMD transform. In – *ASIACRYPT*, pages 299–314, 2006.
11. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS*, 1993.
12. Mihir Bellare and Phillip Rogaway. Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography. In *ASIACRYPT*. Springer, 2000.

13. Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In *EUROCRYPT*, 2006.
14. Mihir Bellare, Asha Camper Singh, Joseph Jaeger, Maya Nyayapati, and Igors Stepanovs. Ratcheted encryption and key exchange: The security of messaging. Cryptology ePrint Archive, Report 2016/1028, 2016. `http://eprint.iacr.org/2016/1028`.
15. Daniel J. Bernstein. ChaCha, a variant of Salsa20. `https://cr.yp.to/chacha/chacha-20080128.pdf`.
16. Daniel J Bernstein. The poly1305-aes message-authentication code. In *International Workshop on Fast Software Encryption*, pages 32–49. Springer, 2005.
17. John Black, Shai Halevi, Hugo Krawczyk, Ted Krovetz, and Phillip Rogaway. Umac: Fast and secure message authentication. In *CRYPTO*, 1999.
18. John Black and Phillip Rogaway. A block-cipher mode of operation for parallelizable message authentication. In *EUROCRYPT*, 2002.
19. John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In *SAC*, 2002.
20. Nikita Borisov, Ian Goldberg, and Eric Brewer. Off-the-record communication, or, why not to use PGP. In *ACM Workshop on Privacy in the Electronic Society*, 2004.
21. Gilles Brassard, David Chaum, and Claude Crépeau. Minimum disclosure proofs of knowledge. *Journal of Computer and System Sciences*, 1988.
22. Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In *CRYPTO*, 1997.
23. Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. In *STOC*, 1996.
24. Ran Canetti, Oxana Poburinnaya, and Mariana Raykova. Optimal-rate non-committing encryption in a CRS model. *IACR Cryptology ePrint Archive*, 2016.
25. Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Improved non-committing encryption with applications to adaptively secure protocols. In *ASIACRYPT*, 2009.
26. Katriel Cohn-Gordon, Cas Cremers, Benjamin Dowling, Luke Garratt, and Douglas Stebila. A formal security analysis of the signal messaging protocol. *IACR ePrint Archive*, 2016.
27. Ivan Damgård and Jesper Buus Nielsen. Improved non-committing encryption schemes based on a general complexity assumption. In *CRYPTO*, 2000.
28. Yevgeniy Dodis, Thomas Ristenpart, John Steinberger, and Stefano Tessaro. To hash or not to hash again?(in) differentiability results for hˆ 2 and hmac. In *–CRYPTO 2012*, 2012.
29. Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. *SIAM Review*, 2003.
30. Facebook. Facebook Messenger app. `https://www.messenger.com/`, 2016.
31. Facebook. Messenger Secret Conversations technical whitepaper. `https://fbnewsroomus.files.wordpress.com/2016/07/secret_conversations_whitepaper-1.pdf`, 2016.
32. Pooya Farshim, Benoît Libert, Kenneth G Paterson, and Elizabeth A Quaglia. Robust encryption, revisited. In *Public-Key Cryptography–PKC 2013*, pages 352–368. Springer, 2013.
33. Pooya Farshim, Claudio Orlandi, and Razvan Rosie. Security of symmetric primitives under incorrect usage of keys. *IACR Transactions on Symmetric Cryptology*, 2017(1):449–473, 2017.

34. Juan A. Garay, Daniel Wichs, and Hong-Sheng Zhou. Somewhat non-committing encryption and efficient adaptively secure oblivious transfer. *IACR Cryptology ePrint Archive*, 2008.
35. Juan A. Garay, Daniel Wichs, and Hong-Sheng Zhou. Somewhat non-committing encryption and efficient adaptively secure oblivious transfer. In *CRYPTO*, 2009.
36. Yitchak Gertner and Amir Herzberg. Committing encryption and publicly-verifiable signcryption. *IACR Cryptology ePrint Archive*, 2003.
37. Shai Halevi and Hugo Krawczyk. Security under key-dependent inputs. In *ACM CCS*, 2007.
38. Carmit Hazay, Arpita Patra, and Bogdan Warinschi. Selective opening security for receivers. In *ASIACRYPT*, 2015.
39. Michael Hearn. Modern anti-spam and E2E crypto. `https://moderncrypto.org/mail-archive/messaging/2014/000780.html`.
40. Brett Hemenway, Rafail Ostrovsky, and Alon Rosen. Non-committing encryption from Φ-hiding. In *TCC (1)*. Springer, 2015.
41. Tetsu Iwata and Kaoru Kurosawa. Omac: One-key cbc mac. In *FSE*, 2003.
42. Hugo Krawczyk. The order of encryption and authentication for protecting communications (or: How secure is ssl?). In *CRYPTO*. Springer, 2001.
43. Feiyu Lei, Wen Chen, and Kefei Chen. A non-committing encryption scheme based on quadratic residue. In *ISCIS*, 2006.
44. Moxie Marlinspike. libsignal protocol (Java). `https://github.com/WhisperSystems/libsignal-protocol-java`, 2016.
45. Moxie Marlinspike and Trevor Perrin. The Double Ratchet algorithm. `https://whispersystems.org/docs/specifications/doubleratchet/doubleratchet.pdf`.
46. David McGrew and John Viega. The galois/counter mode of operation (gcm). *Submission to NIST Modes of Operation Process*, 20, 2004.
47. Jon Millican. Challenges of E2E Encryption in Facebook Messenger. Real World Cryptography conference, 2017. `https://www.realworldcrypto.com/rwc2017/program`.
48. Payman Mohassel. A closer look at anonymity and robustness in encryption schemes. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 501–518. Springer, 2010.
49. Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In *CRYPTO*, 2002.
50. Adam O'Neill, Chris Peikert, and Brent Waters. Bi-deniable public-key encryption. In *CRYPTO*, 2011.
51. Kenneth G Paterson, Thomas Ristenpart, and Thomas Shrimpton. Tag size does matter: Attacks and proofs for the tls record protocol. In *ASIACRYPT*, 2011.
52. Phillip Rogaway. Authenticated-encryption with associated-data. In *ACM CCS*, 2002.
53. Phillip Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes ocb and pmac. In *ASIACRYPT*. Springer, 2004.
54. Phillip Rogaway. Nonce-based symmetric encryption. In *FSE*, 2004.
55. Phillip Rogaway and Thomas Shrimpton. A provable-security treatment of the key-wrap problem. In *EUROCRYPT*, 2006.
56. Mark Dermot Ryan. Enhanced certificate transparency and end-to-end encrypted mail. In *NDSS*. The Internet Society, 2014.
57. Douglas R Stinson. Universal hashing and authentication codes. *Designs, Codes and Cryptography*, 1994.

58. Liang Wang, Rafael Pass, abhi shelat, and Thomas Ristenpart. Secure channel injection and anonymous proofs of account ownership. Cryptology ePrint Archive, Report 2016/925, 2016. `http://eprint.iacr.org/2016/925`.
59. Mark N Wegman and J Lawrence Carter. New hash functions and their use in authentication and set equality. *Journal of computer and system sciences*, 1981.
60. Whatsapp. Whatsapp. `https://www.whatsapp.com/`, 2016.
61. Wikipedia. Signal (software). `https://en.wikipedia.org/wiki/Signal_(software)`, 2016.
62. Huafei Zhu, Tadashi Araragi, Takashi Nishide, and Kouichi Sakurai. Adaptive and composable non-committing encryptions. In *ACISP*, 2010.
63. Huafei Zhu, Tadashi Araragi, Takashi Nishide, and Kouichi Sakurai. Universally composable non-committing encryptions in the presence of adaptive adversaries. In *SECRYPT*, 2010.
64. Huafei Zhu and Feng Bao. Non-committing encryptions based on oblivious naor-pinkas cryptosystems. In *INDOCRYPT*, 2009.
65. Huafei Zhu and Feng Bao. Error-free, multi-bit non-committing encryption with constant round complexity. In *Inscrypt*, 2010.