エンドツーエンド暗号化 SFrame に対する安全性評価

五十部 孝典[1,2,3]　伊藤 竜馬[2,a)]　峯松 一彦[4]

概要：メッセージングアプリケーションやビデオ会議システムの利用拡大に伴い，これらを利用するユーザのプライバシーを保護するためにエンドツーエンド暗号化（E2EE）技術の必要性が高まっている．本稿では，E2EE 技術の 1 つである SFrame に注目する．SFrame は Real-Time Communacation (RTC) 用の E2EE 技術として Google と CoSMo Software により設計され，Google Duo，Cisco Webex，Jitsi Meet で採用予定である．我々は IETF で公開されているインターネットドラフトのバージョン draft-omara-sframe-01 に記載された E2EE 方式に対して安全性評価を実施する．結果として SFrame における認証暗号の使用方法に問題があることを発見し，これらの問題を悪用することで偽造攻撃や認証鍵回復攻撃が現実的な計算量で実行可能であることを明らかにする．さらに，これらの攻撃に対する効果的な対策を提案する．なお，我々の評価結果については SFrame の設計者に連絡済みであり，本稿で示す攻撃が全て実現しうることを確認している．また，SFrame の設計者は我々の脆弱性報告を受けて仕様を修正し，インターネットドラフトのバージョンを draft-omara-sframe-02 に更新している．

キーワード：エンドツーエンド暗号化，SFrame，偽造攻撃，認証鍵回復攻撃

Security Analysis of SFrame

Takanori Isobe[1,2,3]　Ryoma Ito[2,a)]　Kazuhiko Minematsu[4]

Abstract: In this paper, we thoroughly analyse the security of SFrame (version draft-omara-sframe-01), which was designed by a team of Google and CoSMo Software as an end-to-end encryption (E2EE) mechanism for real-time communication (RTC). It is going to be adopted by a number of real-world applications, such as Google Duo, Cisco Webex, and Jitsi Meet. We discover several issues in the authenticated encryption mechanism of SFrame and show forgery and authentication key recovery attacks with practical time complexity by exploiting these issues. In addition, we propose effective countermeasures against these attacks. We reported our results in this paper to the SFrame designers and confirmed that all of the proposed attacks are feasible. The designers modified the specification based on our report and updated the draft version to draft-omara-sframe-02.

Keywords: End-to-End Encryption, SFrame, Forgery, Authentication Key Recovery

1. はじめに

エンドツーエンド暗号化（E2EE）とは，通信相手との間でのみメッセージの送受信が可能であり，通信システムの提供者であってもメッセージの盗聴，改ざんができない暗号化方式のことを指す．2013 年 3 月に発生したスノーデン氏の暴露事件で明らかになったように，インターネットを介した通信では国家規模の大規模な監視活動が行われている危険性がある．このため，ユーザのプライバシーを保護する技術として E2EE の必要性が注目され始めた．実際に，E2EE 技術は Zoom や Webex のようなビデオ会議システムなどで幅広く導入されている．また，E2EE 技術の導入に伴い，E2EE 技術に対する安全性評価についても

[1] 兵庫県立大学，University of Hyogo, Japan.
[2] 国立研究開発法人情報通信研究機構，National Institute of Information and Communications Technology, Japan.
[3] 国立研究開発法人科学技術振興機構，PRESTO, Japan Science and Technology Agency, Japan.
[4] 日本電気株式会社，NEC Corporation, Japan.
[a)] itorym@nict.go.jp

活発に議論されている [5,11,12,13,28].

SFrame はリアルタイム通信が求められるビデオ会議システムなどで効率的な E2EE を実現するために設計された暗号化メカニズムである．2020 年に Google と CoSMo Software のグループ (Omara, Uberti, Gouaillard, Murillo) によって提案され，Google Duo [23]，Cisco Webex [3,4]，Jitsi Meet [14,29] などのアプリケーションで採用される予定である．その最大の特徴は，他のグループコミュニケーションで採用されている E2EE 技術とは異なり，悪意のあるグループメンバーからの偽造攻撃を防ぐために，認証タグに対して計算される各ユーザ固有のデジタル署名を添付していることである．

本稿では，SFrame に対し，第三者として初めての安全性評価を実施する．SFrame の仕様はインターネットドラフトとして公開されており，その最新バージョンは draft-omara-sframe-02 [25] であるが，本稿では直前のバージョンである draft-omara-sframe-01 [24] に記載された E2EE 方式を安全性評価の対象とする．

1.1 本研究の貢献

本研究における最も注目すべき貢献は，認証暗号と署名アルゴリズムの使用に関して重大な欠陥を発見したことである．この重大な欠陥については，ビデオ会議システムなどのグループコミュニケーションの場に悪意のあるグループメンバーが存在する場合に露見するものである．悪意のあるグループメンバーはグループ内で共有する秘密鍵を保有しており，この秘密鍵を悪用することで認証暗号に対して偽造攻撃が可能であることを指摘する．具体的には，

- AES-CM-HMAC（AES-CTR と HMAC-SHA256 の一般的構成）に対し，短いタグ長を指定した場合において現実的な計算量で偽造攻撃が成立することを示す．例えば，タグ長が 4 バイトの場合，オフラインフェーズで 2^{32} 通りの暗号文とタグのペアを保存した事前計算テーブルを作成することにより，100 % に限りなく近い成功確率で偽造攻撃が成立する．
- AES-GCM に対し，任意のタグ長を指定しても現実的な計算量で偽造攻撃が成立することを示す．この攻撃は GHASH が線形関数であることを悪用しており，秘密鍵を保有する攻撃者であれば 100 % の成功確率で偽造攻撃が成立する．AES-CM-HMAC の場合とは異なりオフラインフェーズにおける事前計算が不要であるため，タグ長に依存しない偽造攻撃が可能となる．

SFrame では悪意のあるグループメンバーからの偽造攻撃を防ぐために各ユーザ固有のデジタル署名を添付しているが，我々の指摘は認証タグに対して計算されるデジタル署名を添付したとしても偽造攻撃を防ぐことができない場合があることを明らかにするものである．最後に，我々の提示する攻撃に対して効果的な対策を提案する．

我々の安全性評価はインターネットドラフト [24] と公開されているソースコード [3,14,30] に基づくものであり，我々の提示する攻撃が実際のシステムでうまく機能するとは限らない．また，SFrame の仕様はドラフト版として公開されているだけであり，現時点では実際のシステムに対して差し迫った脅威にはならないと考えている．しかしながら，我々の提示する攻撃の実用性を考えると、インターネットドラフトのバージョン draft-omara-sframe-01 [24] に記載されている SFrame の仕様については改善の余地があると考えている．

1.2 情報開示

本稿で示す評価結果については，2021 年 3 月 にメールとビデオ会議を通じて SFrame の設計者に報告済みであり，公開の許可を得ている．議論を重ねた結果，本稿で示す攻撃については全て実現しうることを確認した．我々の脆弱性報告を受け，SFrame の設計者は署名メカニズムの削除 [9] とタグ生成メカニズムの強化 [8] を速やかに実施し，2021 年 3 月 29 日にインターネットドラフトのバージョンを draft-omara-sframe-02 [25] に更新した．なお，今後は SFrame が署名メカニズムをサポートできるよう対策を講じる予定であるとのことである．

2. SFrame

SFrame はリアルタイム通信が求められるビデオ会議システムなどで効率的な E2EE を実現するために提案された暗号化メカニズムである．その仕様はインターネットドラフト [24] で明記されているが，鍵交換プロトコルについては Signal プロトコル [26]，Olm プロトコル [19]，Message Layer Security (MLS) プロトコル [2] などから実装者が自由に選択できる余地がある．

ユーザは Real-time Transport Protocol (RTP) パケット化に先立ち，SFrame でメディアフレームを暗号化する．その後，RTP パケット化では暗号化されたフレームを 1 つ以上の RTP パケットに分割し，最初のパケットの先頭に SFrame ヘッダを追加し，最後のパケットの末尾に認証タグを追加する．なお，SFrame ヘッダには署名フラグ S，鍵 ID 番号 KID，カウンタ値 CTR が含まれる．

2.1 仕様

本節では，SFrame の仕様について概説する（細部は，文献 [24] を参照されたい）．

全てのグループメンバーは最初に鍵交換プロトコルを実行し，鍵 ID 番号 KID に紐付けられた複数のグループ鍵 $K_{\mathsf{base}}^{\mathsf{KID}}$ を共有する．加えて，各ユーザは署名鍵ペア $(K_{\mathrm{sig}}, K_{\mathrm{verf}})$ を生成する．

SFrame では, 認証暗号として AES-GCM 又は AES-CM-

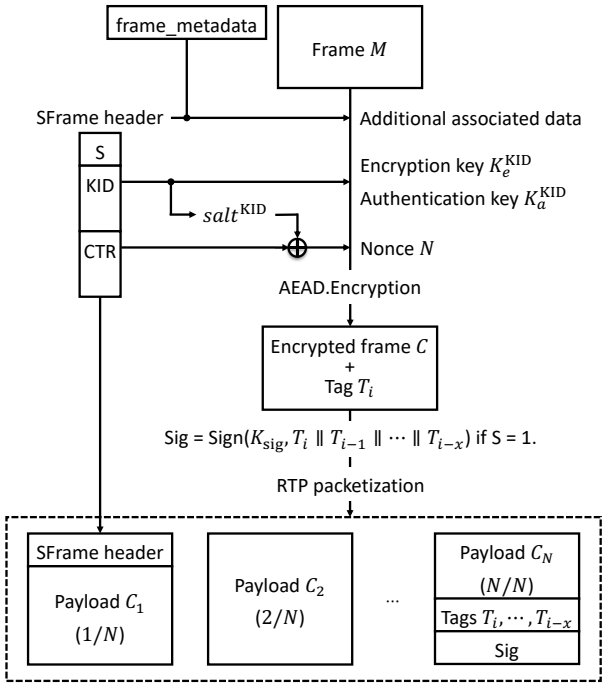

図 1: 暗号プロトコルの全体像

HMAC，ハッシュ関数として SHA256 又は SHA512，署名アルゴリズムとして EdDSA over Ed25519 又は ECDSA over P-521 を採用している．これらの暗号方式から SFrame で使用するための具体的な暗号スイートを構成する．特に，インターネットドラフト [24] では認証暗号方式の細部を次のように定義している．

- AES-GCM の使用においては鍵長を 128 又は 256 ビットとし，タグ長については指定しない．
- AES-CM-HMAC とは AES-CTR と HMAC-SHA256 の組み合わせであり，その使用においては鍵長を 128 ビット，タグ長を 4 又は 8 バイトとする．

図 1 と Alg. 1 は暗号プロトコルの全体像を示している．暗号スイートとして AES-GCM を利用する場合，Alg. 1 における AEAD.Encryption は NIST SP 800-38D [7] に従って実行される．なお，AES-GCM の実行に先立ち，暗号化鍵 K_e^{KID} とソルト $salt^{\mathsf{KID}}$ は HKDF [18] を使用して次のように生成される．

$$\begin{aligned}\mathsf{SFrameSecret} &= \mathsf{HKDF}(K_{\mathsf{base}}^{\mathsf{KID}}, \text{'SFrame10'}),\\ K_e^{\mathsf{KID}} &= \mathsf{HKDF}(\mathsf{SFrameSecret}, \text{'key'}, \mathrm{KeyLen}),\\ salt^{\mathsf{KID}} &= \mathsf{HKDF}(\mathsf{SFrameSecret}, \text{'salt'}, \mathrm{NonceLen}).\end{aligned}$$

ここで，KeyLen は鍵長，NonceLen はナンス長を表す．その後，各ユーザは K_e^{KID} と $salt^{\mathsf{KID}}$ を KeyStore[KID] に保存する．一方，暗号スイートとして AES-CM-HMAC を利用する場合，Alg. 1 における AEAD.Encryption は Alg. 2 に従って実行される．なお，AES-CM-HMAC の実行に先立ち，暗号化鍵 K_e^{KID}，認証鍵 K_a^{KID}，ソルト $salt^{\mathsf{KID}}$ が HKDF [18] を使用して次のように生成される．

$$\begin{aligned}\mathsf{AEADSecret} &= \mathsf{HKDF}(K_{\mathsf{base}}^{\mathsf{KID}}, \text{'SFrame10 AES CM AEAD'}),\\ K_e^{\mathsf{KID}} &= \mathsf{HKDF}(\mathsf{AEADSecret}, \text{'key'}, \mathrm{KeyLen}),\\ K_a^{\mathsf{KID}} &= \mathsf{HKDF}(\mathsf{AEADSecret}, \text{'auth'}, \mathrm{HashLen}),\\ salt^{\mathsf{KID}} &= \mathsf{HKDF}(\mathsf{AEADSecret}, \text{'salt'}, \mathrm{NonceLen}),\end{aligned}$$

ここで，HashLen はハッシュ関数の出力長を表す．その後，各ユーザは K_e^{KID}，K_a^{KID}，$salt^{\mathsf{KID}}$ を KeyStore[KID] に保存する．

認証暗号は $K_{\mathsf{base}}^{\mathsf{KID}}$ を保有しない攻撃者に対して偽造を検知することができるものの，$K_{\mathsf{base}}^{\mathsf{KID}}$ を保有するグループメンバーが悪意を持って実行する偽造攻撃を防ぐことはできない．このような攻撃に対しては暗号化されたパケットに署名を添付することが有効な対策の 1 つであり，効率性の観点から SFrame ではタグのリスト $(T_i, T_{i-1}, \ldots, T_{i-x})$ に対して次のように署名 Sig を計算する．

$$\mathsf{Sig} = \mathsf{Sign}(K_{\mathrm{sig}}, T_i \parallel T_{i-1}, \parallel \cdots \parallel T_{i-x}).$$

Algorithm 1 暗号プロトコル

Input: S: 署名フラグ, KID: 鍵 ID 番号, CTR: カウンタ値, frame_metadata: フレームメタデータ, M: フレーム
Output: C: 暗号化フレーム, T: 認証タグ

procedure Encryption(S, KID, CTR, frame_metadata, M)
  if An AEAD encryption algorithm is AES-GCM **then**
    $K_e^{\mathsf{KID}}, salt^{\mathsf{KID}} = \mathsf{KeyStore}[\mathsf{KID}]$
  else
    $K_e^{\mathsf{KID}}, K_a^{\mathsf{KID}}, salt^{\mathsf{KID}} = \mathsf{KeyStore}[\mathsf{KID}]$
  end if
  $ctr = \mathsf{encode}(\mathsf{CTR}, \mathrm{NonceLen})$
  $N = salt^{\mathsf{KID}} \oplus ctr$
  $\mathsf{header} = \mathsf{encode}(\mathsf{S}, \mathsf{KID}, \mathsf{CTR})$
  $\mathsf{aad} = \mathsf{header} + \mathsf{frame\_metadata}$
  if an AEAD encryption algorithm is AES-GCM **then**
    $C, T = \text{AEAD.Encryption}(K_e^{\mathsf{KID}}, N, \mathsf{aad}, M)$
  else
    $C, T = \text{AEAD.Encryption}(K_e^{\mathsf{KID}}, K_a^{\mathsf{KID}}, N, \mathsf{aad}, M)$
  end if
end procedure

Algorithm 2 AES-CM-HMAC による暗号化とタグ生成

Input: K_a^{KID}: 認証鍵, aad: 関連データ, C: 暗号化フレーム
Output: T: 認証タグ

procedure Tag.Generation(K_a^{KID}, aad, C)
  $\mathsf{aad}Len = \mathsf{encode}(len(\mathsf{aad}), 8)$
  $D = \mathsf{aad}Len + \mathsf{aad} + C$
  $tag = \mathsf{HMAC}(K_a^{\mathsf{KID}}, D)$
  $T = \mathsf{trancate}(tag, \mathrm{TagLen})$
end procedure

Input: K_e^{KID}, K_a^{KID}, N: ナンス, aad, M: フレーム
Output: C, T

procedure AEAD.Encryption(K_e^{KID}, K_a^{KID}, N, aad, M)
  $C = \text{AES-CTR.Encryption}(K_e^{\mathsf{KID}}, N, M)$
  $T = \text{Tag.Generation}(K_a^{\mathsf{KID}}, \mathsf{aad}, C)$
end procedure

ここで，Sign は署名アルゴリズムを表す．

2.2 実装例

本節では，公開されている SFrame の実装例を紹介する．

オリジナル実装．SFrame の設計者の 1 人である Sergio Garcia Murillo によって実装されたソースコードが公開されている [30]．この実装では，タグ長を 4 又は 10 バイトとした AES-CM-HMAC をサポートしている．

Google Duo．Duo は Google によって開発されたビデオ通話アプリケーションであり，鍵交換メカニズムとして Signal プロトコル，E2EE メカニズムとして SFrame を採用している．SFrame の設計者の 1 人である Emad Omara によって執筆されたテクニカルペーパー [23] が公開されているが，ソースコードは公開されていない．テクニカルペーパー [23] によると AES-CM-HMAC をサポートしているが，タグ長は不明である．なお，現状では署名アルゴリズムを使用していないことが確認できている．

Cisco Webex．Webex は Cisco によって開発されたビデオ会議システムであり，鍵交換メカニズムとして MLS プロトコル，E2EE メカニズムとして SFrame を採用している．SFrame の実装は Github [3] で公開されており，鍵長を 128 又は 256 ビット，タグ長を 16 バイトとした AES-GCM とタグ長を 4 又は 8 バイトとした AES-CM-HMAC をサポートしている．

Jitsi Meet．Jitsi Meet は FOSDEM 2021[*1] で紹介されたビデオ通話アプリケーションであり，鍵交換メカニズムとして Olm プロトコル，E2EE メカニズムとして SFrame を採用している．SFrame の実装は Github [14] で公開されており，タグ長を 4 又は 10 バイトとした AES-CM-HMAC をサポートしている．

3. 攻撃者モデルと安全性要件

3.1 攻撃者モデル

Isobe と Minematsu [13] によって定義された**悪意のあるユーザ** (**Malicious User**) と**悪意のあるグループメンバー** (**Malicious Group Member**) を攻撃者モデルとして設定する．細部は，文献 [13] を参照されたい．

3.2 E2EE における安全性要件

2021 年 2 月に公開されたインターネットドラフト [16] で定義された**機密性** (**Confidentiality**)，**完全性** (**Integrity**)，そして**真正性** (**Authenticity**) を E2EE における安全性要件として設定する．細部は，文献 [16] を参照されたい．

3.3 E2EE における認証暗号の安全性要件

Dodis ら [6] は，message franking 技術で利用可能な *encryptment* と呼ばれる新しいプリミティブを提案した．加えて，このプリミティブの安全性要件として *second-ciphertext unforgeability* (SCU) を定義した．

定義 1. (Second-Ciphertext Unforgeability (SCU)) 攻撃者 $\mathcal{A}$ は，鍵空間 $\mathcal{K}$ からランダムに選ばれた秘密鍵 K が与えられ，ローカル環境で認証暗号の暗号化と復号を実行できると仮定する．この時，*E2EE* における認証暗号に対し，$\mathcal{A}$ の *SCU advantage* は次のとおり．

$$\begin{aligned}&\mathbf{Adv}^{\mathsf{SCU}}_{\mathsf{AEAD}}(\mathcal{A})\\ &= \Pr\big[K \xleftarrow{\$} \mathcal{K} : \mathcal{A}(K) \to (N, A, C, N^*, A^*, C^*, T),\\ &\qquad \mathsf{Dec}(K, N, A, C, T) = M,\\ &\qquad \mathsf{Dec}(K, N^*, A^*, C^*, T) = M^* \textit{ for some } M, M^* \neq \perp\big].\end{aligned}$$

ここで，*Dec* は認証暗号の復号アルゴリズム，N と N^* はナンス，A と A^* は関連データ，C と C^* は暗号文，M と M^* は平文，T はタグ，そして $\perp$ は復号失敗を表す．

E2EE における悪意のあるグループメンバーはグループ鍵 K が共有されているため，ターゲットとなるメディアフレーム (N, A, C, T) を傍受することで SCU 攻撃者 $\mathcal{A}$ として機能することに注意されたい．

3.4 ハッシュ関数の安全性要件

ハッシュ関数 H は一般的に**原像計算困難性**，**第 2 原像計算困難性**，**衝突耐性**という性質を満たす必要がある．ここでは，文献 [21, 27] で定義された 2 つの第 2 原像計算困難性 (Second-Preimage Resistance) に着目する．

定義 2. (Second-Preimage Resistance) 攻撃者 $\mathcal{A}$ は，平文空間 $\mathcal{M}$ からランダムに選ばれた平文 M が与えられ，$H(M) = H(M^*)$ かつ $M \neq M^*$ を満たす平文 M^* を計算しようとする．この時，H に対し，$\mathcal{A}$ の *second-preimage (Sec) resistance advantage* は次のとおり．

$$\begin{aligned}\mathbf{Adv}^{\mathsf{Sec}}_{H}(\mathcal{A}) = \Pr\big[&M \xleftarrow{\$} \mathcal{M}; M^* \leftarrow \mathcal{A} :\\ &(M \neq M^*) \wedge \big(H(M) = H(M^*)\big)\big].\end{aligned}$$

定義 3. (Everywhere Second-Preimage Resistance) 自然数 n において，$\{0,1\}^{\leq n}$ を n ビット以下のビット列とし，$\mathcal{M} = \{0,1\}^*$ と $\mathcal{Y} = \{0,1\}^n$ とする．また，$H : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$ を鍵付きハッシュ関数とする．ここで，攻撃者 $\mathcal{A}$ は，平文空間 $\mathcal{M}$ からランダムに選ばれた平文 M ($|M| \leq \ell$) が与えられ，$H(K, M) = H(K, M^*)$ かつ $M \neq M^*$ を満たす平文 M^* を計算しようとする．この時，H に対し，$\mathcal{A}$ の *everywhere second-preimage (eSec) resistance advantage* は次のとおり．

$$\begin{aligned}\mathbf{Adv}^{\mathsf{eSec}[\leq \ell]}_{H}(\mathcal{A}) = \max_{M \in \{0,1\}^{\leq \ell}} \Big\{ \Pr\big[&K \xleftarrow{\$} \mathcal{K}; M^* \leftarrow \mathcal{A}(K) :\\ &(M \neq M^*) \wedge \big(H_K(M) = H_K(M^*)\big)\big] \Big\}.\end{aligned}$$

[*1] https://fosdem.org/2021/schedule/

4. 安全性評価

4.1 SFrame で使用する認証暗号の安全性

本節では，SFrame で使用する認証暗号の安全性について説明する．仕様の細部は，第 2.1 節のとおり．

Alg. 1 に注目すると，変数 N は認証暗号アルゴリズムへの入力の 1 つであるナンスとして機能し，$salt^{\mathsf{KID}}$ と ctr の排他的論理和から得られる．$salt^{\mathsf{KID}}$ はグループ鍵 $K^{\mathsf{KID}}_{\mathsf{base}}$ から HKDF を用いて生成され，ctr はカウンタ値 CTR をエンコードした値である．また，データ aad も同様に関連データとして機能し，header と frame_metadata から得られる．header は $(\mathsf{S}, \mathsf{KID}, \mathsf{CTR})$ をエンコードした値である．aad には N と同様に CTR が含まれているため，認証暗号として AES-CM-HMAC を使用する場合，HMAC の入力には関連データ aad と暗号文 C に加えてナンスとして機能する CTR が含まれていることになる．つまり，HMAC の入力にナンス $N = salt^{\mathsf{KID}} \oplus ctr$ が含まれていないことは安全性の観点で問題ないと言える．

さらに詳細に分析すると，Alg. 2 で示される AES-CM-HMAC は encrypt-then-MAC 構造の認証暗号であると解釈できる．具体的に示すと，これはナンス $\widetilde{N} = \mathsf{CTR}$，関連データ $\widetilde{A} = (\mathsf{S}, \mathsf{KID}, \mathsf{frame\_metadata})$，平文 M を入力とし，次の手順で 暗号文 C とタグ T を生成する．

$$C = \widetilde{\mathsf{Enc}}_K(\widetilde{N}, M)$$
$$T = \widetilde{\mathsf{MAC}}_{K'}(\widetilde{N}, \widetilde{A}, C),$$

ここで，K と K' はグループ鍵 $K^{\mathsf{KID}}_{\mathsf{base}}$ から HKDF を用いて生成される．また，$\widetilde{\mathsf{Enc}}_K$ はナンスへ擬似ランダムオフセットを適用したカウンタモードによる暗号化，$\widetilde{\mathsf{MAC}}_{K'}$ は入力へ全単射性を有する符号化を適用した HMAC を表す．これは Alg. 1 が encrypt-then-MAC の一般的構成に帰着されることを意味する．この一般的構成における安全性については $\widetilde{\mathsf{Enc}}$ が IND-CPA 安全，かつ，$\widetilde{\mathsf{MAC}}$ が擬似ランダム関数である場合において証明されており [17, 22]，$\widetilde{\mathsf{MAC}}$ が擬似ランダム関数であるという主張は容易に証明可能である．つまり，Alg. 1 は AES が擬似ランダム置換であり，かつ，HMAC が 擬似ランダム関数であるという一般的な仮定の下で安全であると言える．

なお，ナンス N と関連データ aad が独立である場合，Alg. 2 ではタグ生成に N が含まれていないため，その一般的構成を考えた場合に安全性が損なわれることに注意されたい．この問題点については CFRG[*2] で議論され，その解決策として本節での評価結果を提示している．

4.2 短いタグを出力する AES-CM-HMAC の安全性

本節では，グループ鍵を保有する悪意のあるグループメンバーによって偽造攻撃が可能であることを説明する．具体的には，SFrame で短いタグを出力する AES-CM-HMAC を採用する場合，SFrame が E2EE における安全性要件を満たさないことを示す．

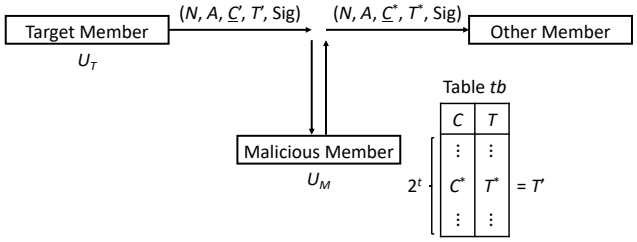

図 2: 短いタグを出力する AES-CM-HMAC への偽造攻撃

以降，標準的な認証暗号における表記を用いて簡略化モデルを構成する．つまり，認証暗号の入出力 (N, A, M, C, T) をそれぞれ ナンス N，関連データ A，平文 M，暗号文 C，そしてタグ T と表記する．Alg. 1 と Alg. 2 で示した手順と同様に，簡略化モデルにおいて各グループメンバーは他の全てのメンバーに暗号化フレームを送信する．この暗号化フレームには (N, A, C, T) と 送信者の署名鍵 K_{sig} で計算されたタグに対する署名 Sig が含まれる．AES-CM-HMAC によるフレームの暗号化手順は，次のとおり．

$$C \leftarrow \text{AES-CTR}(K^{\mathsf{KID}}_e, N, M),$$
$$T \leftarrow \mathsf{truncate}(\text{HMAC-SHA256}(K^{\mathsf{KID}}_a, (N, A, C)), \tau).$$

ここで，τ はタグのビット長を表す．

グループ内に悪意のあるグループメンバー U_M とターゲットメンバー U_T が存在すると仮定する．U_M は U_T から送信された暗号化フレームを傍受し，暗号文を差し替えることによって偽造攻撃が実行可能となる．我々の提案する偽造攻撃はオフラインフェーズとオンラインフェーズで構成され，U_M は次の攻撃手順を実行する．なお，攻撃の全体像は図 2 のとおりである．

オフラインフェーズ．

1. 認証暗号への入力 (N, A, M) を選択する．
2. 選択した (N, A, M) に対応する暗号文 C と τ ビットのタグ T を計算する．
3. (C, T) ペアを事前計算テーブル tb に保存する．
4. ステップ 1–3 を 2^t 回繰り返し実行する．

オンラインフェーズ．

1. U_T から送信された暗号化フレーム $(N, A, C', T', \mathsf{Sig})$ を傍受する．
2. 事前計算テーブルから $T^* = T'$ かつ $C^* \neq C'$ となる (C^*, T^*) ペアを探索する．
3. 該当するペアが存在する場合，暗号化フレーム内の C' を C^* に差し替えたフレーム $(N, A, C^*, T', \mathsf{Sig})$ を他のグループメンバーに送信する．

[*2] https://mailarchive.ietf.org/arch/browse/cfrg/?q=SFrame

Sig は T' に対して計算されることに注意されたい．U_M によって偽造された暗号化フレームには偽造前と同じ T' が含まれているため，タグ検証だけでなく署名検証についても問題なく成功する．つまり，グループメンバーは U_M によって暗号化フレームが偽造されたことを検知できず，偽造された暗号文 C^* の復号結果 M^* が U_T から送信された平文として正式に受理されることとなる．

4.2.1 計算量評価

オフラインフェーズで事前計算テーブル tb を作成する計算量を 2^t と見積もった場合，オンラインフェーズで攻撃が成功する確率を $2^{-\tau+t}$ と見積もることができる．

4.2.2 攻撃の実現可能性

SFrame ヘッダにはリプレイ攻撃を防ぐためのフレームカウンタが含まれている．このため，悪意のあるグループメンバーはオフラインフェーズで (C^*, T^*) ペアを計算する際にターゲットとするフレームカウンタを事前に決定し，その値を SFrame ヘッダとして設定する必要がある．

タグ長が 4 バイト，つまり $\tau = 32$ において，U_M が 2^{32} 通りの (C^*, T^*) ペアを保存した事前計算テーブルを作成する場合，攻撃成功確率は 1 となる．つまり，タグ長が 4 バイトの認証暗号を採用する場合，現実的な計算量で偽造攻撃が実行可能となる．加えて，攻撃者は偽造された暗号化フレームに含まれる C^* の復号結果 M^* を完全にコントロールできるため，任意の平文へと偽造することが可能である．また，タグ長が 8 バイト，つまり $\tau = 64$ であっても，U_M が国家レベルの攻撃者であり，約 2^{56} 通りの (C^*, T^*) ペアを保存した事前計算テーブルを作成できると仮定すると，約 2^{-8} の成功確率で偽造攻撃が可能となる．

4.3 長いタグを出力する AES-CM-HMAC の安全性

本節では，16 バイトタグのような長いタグを出力する AES-CM-HMAC の安全性について説明する．

まず，第 4.2 節で提案した偽造攻撃について考える．タグ長が 16 バイト，つまり $\tau = 128$ の場合，悪意のあるグループメンバーが国家レベルの攻撃者であり，約 2^{56} 通りの (C^*, T^*) ペアを保存した事前計算テーブルを作成できると仮定しても，成功確率は約 2^{-72} となり，現実的な偽造攻撃が可能であるとは言えない．つまり，第 4.2 節で提案した偽造攻撃に対し，長いタグを出力する AES-CM-HMAC は安全であると言える．

次に，AES-CM-HMAC の SCU 安全性を示すことで，上記の主張に関する正当性を示す．Alg. 2 の手順に従い，$D = (N, A, C)$ と $D^* = (N^*, A^*, C^*)$ とすると，HMAC によるタグ生成は次のとおり．

$$\mathrm{HMAC}(K_a^{\mathsf{KID}}, D) = H\big((K \oplus opad) \parallel H\big((K \oplus ipad) \parallel D\big)\big).$$

ここで，H はハッシュ関数，$ipad$ と $opad$ は仕様で定められているパディング値，K は HMAC におけるパディングルールに従って K_a^{KID} から生成された値を表す（細部は，文献 [31] を参照されたい）．この時，AES-CM-HMAC の SCU 安全性は次のとおり．

定理 1. *$\mathcal{A}$ を AES-CM-HMAC に対する SCU 攻撃者とする．この時，AES-CM-HMAC に対する $\mathcal{A}$ の SCU advantage は，H に対していかなる eSec 攻撃者 $\mathcal{A}'$ が存在する場合においても，以下の不等式が成立する．*

$$\mathbf{Adv}^{\mathsf{SCU}}_{\text{AES-CM-HMAC}}(\mathcal{A}) < 2\mathbf{Adv}^{\mathsf{eSec}[\le(\ell')]}_{H}(\mathcal{A}').$$

ここで，ℓ' はハッシュ関数への入力ビット長 ℓ に *1* ブロック分のビット長を追加した値を表す．つまり，ハッシュ関数が *SHA256* の場合，$\ell' = \ell + 512$ となる．

紙面の都合上，定理 1 の証明は割愛する．定理 1 は，AES-CM-HMAC の SCU 安全性がその基礎となるハッシュ関数の安全性に依存することを示している．インターネットドラフト [24] によると，SFrame は AES-CM-HMAC で使用するハッシュ関数として SHA-256 を指定している．

4.3.1 SHA256 の第 2 原像計算困難性

一般的に，n ビットのハッシュ関数は第 2 原像攻撃に対して n ビットのセキュリティレベルを有している．つまり，SHA256 に対する第 2 原像攻撃は 2^{256} の計算量で実行可能であることを意味している．また，SHA256 に対する第 2 原像攻撃は Khovratovich ら [15] や Andreeva ら [1] などによって提案されているものの，いずれも現実的な計算量で実行可能であるとは言えない．

以上より，長いタグを出力する AES-CM-HMAC は SCU 安全な認証暗号であると言える．

4.4 任意長のタグを出力する AES-GCM の安全性

本節では，任意長のタグを出力する AES-GCM の安全性について説明する．第 4.2 節で説明した偽造攻撃は E2EE における認証暗号への汎用的な攻撃であり，オフラインフェーズにおける計算量はタグ長に依存する．一方，AES-GCM の場合はオフラインフェーズを実行する必要が無く，タグ長に依存しない偽造攻撃が実行可能となる．

グループ内に悪意のあるグループメンバー U_M が存在すると仮定する．U_M はグループ鍵を保有しており，通信を傍受することで GCM への正当な入出力 (N, A, C, T) を入手可能である．また，GHASH は線形関数である．つまり，グループ鍵を知っている U_M は $(N', A', C') \neq (N, A, C)$ かつ $T' = T$ となるような GCM の入出力 (N', A', C', T') を自由に生成することができることを意味する．

ここで，ナンス長が 96 ビット，タグ長が 128 ビットである AES-GCM の場合で具体例を示す．2 ブロックの平文 $M = (M_1, M_2)$ と 1 ブロックの関連データ $A = A_1$ で構成される AES-GCM への入力 (N, A, M) が与えられる場合，次の手順で $C = (C_1, C_2)$ と T が計算される．

$$
\begin{aligned}
T &= \mathsf{GHASH}(L, A \,\|\, C \,\|\, \mathsf{len}(A,C)) \oplus E_K(N \,\|\, 1_{32}) \\
&= A{\cdot}L^4 \oplus C_1{\cdot}L^3 \oplus C_2{\cdot}L^2 \oplus \mathsf{len}(A,C){\cdot}L \oplus E_K(N \,\|\, 1_{32}), \\
C_1 &= E_K(N \,\|\, 2_{32}) \oplus M_1, \\
C_2 &= E_K(N \,\|\, 3_{32}) \oplus M_2.
\end{aligned}
$$

ここで，乗算は GF(2^{128}) 上で実行され，$\mathsf{len}(A,C)$ は A と C のビット長，$E_K(*)$ は グループ鍵から導出される秘密鍵 K での AES による暗号化，$L = E_K(0^{128})$ は認証鍵，i_{32} は非負整数 i の 32 ビットエンコーディングを表す．

U_M は K を知っているため，認証鍵 L を容易に計算できることに注意されたい．U_M は最初に任意の N' と A' を選択する．次に，選択した N' と偽造した平文ブロック M_1' から暗号文ブロック C_1' を計算する．最終的に，以下の等式が成り立つように C_2' を決定する．

$$
C_2'{\cdot}L^2 = T' \oplus A'{\cdot}L^4 \oplus C_1'{\cdot}L^3 \oplus \mathsf{len}(A',C'){\cdot}L \oplus E_K(N' \,\|\, 1_{32})
$$

ここで，$T' = T$ とすると，U_M によって偽造された暗号化フレームには偽造前と同じタグが含まれることになるため，タグ検証だけでなく署名検証についても問題なく成功する．なお，C_2' に対応する（最終の）平文ブロック M_2' は任意の値に偽造できないことに注意されたい．

4.4.1 攻撃の実現可能性

上述のとおり，悪意のあるグループメンバーは最終の平文ブロックを除いて任意の平文に偽造することができる．また，この攻撃はタグ長に依存せず実行可能である．

グループ通話における平文とはビデオやオーディオなどのデータであるため，最終の平文ブロックがランダムなデータであったとしてもノイズとして認識される可能性が高い．つまり，本節で提案した攻撃は現実的な計算量で実行可能であるとともに，グループ通話における完全性を著しく損なっていると言える．

4.5 短いタグを出力する AES-GCM への認証鍵回復

インターネットドラフト [24] によると，AES-GCM を使用する場合でも 4 又は 8 バイトのような短いタグ長を指定することが可能である．本節では，短いタグを出力する AES-GCM の安全性について説明する．

短いタグを出力する AES-GCM を使用する場合，前節で指摘した攻撃に加え，悪意のあるユーザによる認証鍵回復攻撃のリスクを考慮する必要がある．なお，認証鍵の導出によって普遍的偽造攻撃も可能となる．

この攻撃の実行可能性については Ferguson [10] によって初めて指摘された．さらに，Mattsson と Westerlund [20] は Ferguson の研究 [10] を発展させ，攻撃にかかる計算量の厳密な評価を実施した．細部は，表 1 のとおり．表 1 から，NIST が定める制約事項 (L,q) [7] を厳守する場合，32 ビットタグを出力する GCM への認証鍵回復攻撃に対して少なくとも 61 ビットのセキュリティレベルを有していることが明らかである．

表 1: t ビットタグを出力する GCM 使用時の制約事項 (L,q) [7] と認証鍵回復攻撃にかかる計算量 c の評価結果 [20]．L は暗号文と関連データを組み合わせたバイト長の最大値，q は GHASH 関数の最大呼び出し回数を表す．

t	32						64					
L	2^1	2^2	2^3	2^4	2^5	2^6	2^{11}	2^{13}	2^{15}	2^{17}	2^{19}	2^{21}
q	2^{22}	2^{20}	2^{18}	2^{15}	2^{13}	2^{11}	2^{32}	2^{29}	2^{26}	2^{23}	2^{20}	2^{17}
c	2^{62}	2^{62}	2^{61}	2^{65}	2^{66}	2^{67}	2^{75}	2^{74}	2^{73}	2^{72}	2^{71}	2^{70}

一方，表 1 で示す制約事項 (L,q) を厳守しない場合，認証鍵は 2^t の計算量で復元される．これは 32 ビットタグを出力する GCM への認証鍵回復攻撃が 2^{32} の計算量で実行可能であることを意味する．

4.5.1 攻撃の実現可能性

インターネットドラフト [24] には制約事項 (L,q) が明記されていない．つまり，仕様 [24] に基づいて 32 ビットタグを出力する AES-GCM を実装する場合，制約事項が考慮されない可能性が高いと考えられる．さらに，この実装では認証鍵回復攻撃に対して 32 ビットのセキュリティレベルしか満たさないため，悪意のあるユーザは現実的な計算量で認証鍵回復攻撃を実行できることになる．

4.6 対策

以上の安全性評価を踏まえ，次のような対策を提案する．

- 短いタグ長，特に 4 バイトのタグを出力する AES-CM-HMAC を使用しない．
- AES-GCM を使用する場合，タグだけではなくフレーム全体に対して署名を計算する．
- 短いタグを出力する AES-GCM の使用を禁止する，又は NIST が示す制約事項を仕様に明記する．
- 暗号スイートを構成する暗号方式の 1 つとして HFC [6] のような安全な encryptment 方式を採用する．

5. まとめ

本研究では，E2EE 技術の 1 つである SFrame に対してインターネットドラフトのバージョン draft-omara-sframe-01 [24] に基づき安全性評価を実施した．SFrame では悪意のあるグループメンバーからの偽造攻撃を防ぐために各ユーザ固有のデジタル署名を添付しているが，認証タグに対して計算されるデジタル署名を添付したとしても偽造攻撃を防ぐことができない場合があることを明らかにした．また，長いタグ長を出力する AES-CM-HMAC の証明可能安全性を示すとともに，この認証暗号が我々の提示する偽造攻撃への効果的な対策となりうることを明らかにした．

本稿で示す評価結果については SFrame の設計者に連絡済みであり，我々の指摘する攻撃については全て実現し

うることを確認した．また，我々の脆弱性報告を受けて設計者は速やかに SFrame の仕様を修正した．SFrame がグループコミュニケーション用の E2EE 技術として幅広く導入されることを考えると，SFrame に対する安全性評価はさらに活発に行われるべきであり，我々の研究がその発展に貢献できることを願っている．

謝辞

SFrame の設計者 (Emad Omara, Justin Uberti, Alex Gouaillard, Sergio Garcia Murillo) と株式会社時雨堂から本研究に対する有意義な議論とアドバイスをいただきました．この場を借りて感謝申し上げます．

参考文献

[1] Andreeva, E., Bouillaguet, C., Dunkelman, O., Fouque, P., Hoch, J.J., Kelsey, J., Shamir, A., Zimmer, S.: New Second-Preimage Attacks on Hash Functions. J. Cryptol. **29**(4), 657–696 (2016)

[2] Barnes, R., Beurdouche, B., Millican, J., Omara, E., Cohn-Gordon, K., Robert, R.: The Messaging Layer Security (MLS) Protocol. https://tools.ietf.org/html/draft-ietf-mls-protocol-10 (October 2020)

[3] Cisco Systems: SFrame (2020), https://github.com/cisco/sframe

[4] Cisco Systems: Zero-Trust Security for Webex White Paper (2021), https://www.cisco.com/c/en/us/solutions/collateral/collaboration/white-paper-c11-744553.pdf

[5] Cohn-Gordon, K., Cremers, C., Dowling, B., Garratt, L., Stebila, D.: A Formal Security Analysis of the Signal Messaging Protocol. J. Cryptol. **33**(4), 1914–1983 (2020)

[6] Dodis, Y., Grubbs, P., Ristenpart, T., Woodage, J.: Fast Message Franking: From Invisible Salamanders to Encryptment. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10991, pp. 155–186. Springer (2018)

[7] Dworkin, M.:

[8] Emad Omara: Extend Tag Calculation to Cover Nonce #59 (2021), https://github.com/eomara/sframe/pull/59

[9] Emad Omara: Remove Signature #58 (2021), https://github.com/eomara/sframe/pull/58

[10] Ferguson, N.: Authentication Weaknesses in GCM. Comments submitted to NIST Modes of Operation Process (2005), http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/comments/CWC-GCM/Ferguson2.pdf

[11] Garman, C., Green, M., Kaptchuk, G., Miers, I., Rushanan, M.: Dancing on the Lip of the Volcano: Chosen Ciphertext Attacks on Apple iMessage. In: Holz, T., Savage, S. (eds.) USENIX Security 2016. pp. 655–672. USENIX Association (2016)

[12] Isobe, T., Ito, R.: Security Analysis of End-to-End Encryption for Zoom Meetings. IEEE Access **9**, 90677–90689 (2021)

[13] Isobe, T., Minematsu, K.: Breaking message integrity of an end-to-end encryption scheme of LINE. In: López, J., Zhou, J., Soriano, M. (eds.) ESORICS 2018. LNCS, vol. 11099, pp. 249–268. Springer (2018)

[14] Jitsi: Jitsi Meet API library (2020), https://github.com/jitsi/lib-jitsi-meet/

[15] Khovratovich, D., Rechberger, C., Savelieva, A.: Bicliques for Preimages: Attacks on Skein-512 and the SHA-2 Family. In: Canteaut, A. (ed.) FSE 2012. LNCS, vol. 7549, pp. 244–263. Springer (2012)

[16] Knodel, M., Baker, F., Kolkman, O., Celi, S., Grover, G.: Definition of End-to-end Encryption. https://datatracker.ietf.org/doc/draft-knodel-e2ee-definition/ (February 2021)

[17] Krawczyk, H.: The Order of Encryption and Authentication for Protecting Communications (or: How Secure Is SSL?). In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 310–331. Springer (2001)

[18] Krawczyk, H., Eronen, P.: HMAC-based Extract-and-Expand Key Derivation Function (HKDF). Internet Engineering Task Force - IETF, Request for Comments **5869** (May 2010)

[19] Matrix.org Foundation.: Olm: A Cryptographic Ratchet (2016), https://gitlab.matrix.org/matrix-org/olm/-/blob/master/docs/olm.md

[20] Mattsson, J., Westerlund, M.: Authentication Key Recovery on Galois/Counter Mode (GCM). In: Pointcheval, D., Nitaj, A., Rachidi, T. (eds.) AFRICACRYPT 2016. LNCS, vol. 9646, pp. 127–143. Springer (2016)

[21] Menezes, A.J., Oorschot, P.C.V., Vanstone, S.A.: Handbook of Applied Cryptography. CRC press (1996)

[22] Namprempre, C., Rogaway, P., Shrimpton, T.: Reconsidering Generic Composition. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 257–274. Springer (2014)

[23] Omara, E.: Google Duo End-to-End Encryption Overview - Technical Paper (2020), https://www.gstatic.com/duo/papers/duo_e2ee.pdf

[24] Omara, E., Uberti, J., Gouaillard, A., Murillo, S.G.: Secure Frame (SFrame). https://tools.ietf.org/html/draft-omara-sframe-01 (November 2020)

[25] Omara, E., Uberti, J., Gouaillard, A., Murillo, S.G.: Secure Frame (SFrame). https://tools.ietf.org/html/draft-omara-sframe-02 (March 2021)

[26] Open Whisper Systems.: Signal Github Repository (2017), https://github.com/WhisperSystems/

[27] Rogaway, P., Shrimpton, T.: Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. In: Roy, B.K., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 371–388. Springer (2004)

[28] Rösler, P., Mainka, C., Schwenk, J.: More is Less: On the End-to-End Security of Group Chats in Signal, WhatsApp, and Threema. In: 2018 IEEE European Symposium on Security and Privacy (EuroS&P). pp. 415–429. IEEE (2018)

[29] Saĺ Ibarra Corretgé: The road to End-to-End Encryption in Jitsi Meet (2021), https://fosdem.org/2021/schedule/event/e2ee/attachments/slides/4435/export/events/attachments/e2ee/slides/4435/E2EE.pdf

[30] Sergio Garcia Murillo: SFrame.js (2020), https://github.com/medooze/sframe

[31] Turner, J.M.: The Keyed-Hash Message Authentication Code (HMAC). Federal Information Processing Standards Publication **198**, 1 (2008)