Secure Media Frames
Internet-Draft
Updates: 9605 (if approved)
Intended status: Standards Track
Expires: 29 October 2026

R. Barnes
Cisco
E. Omara
A. Rosenberg
Apple

27 April 2026

Updates to SFrame Cipher Suites Registry
draft-barnes-sframe-iana-256-05

Abstract

This document addresses two omissions in the Secure Frames (SFrame) protocol specification. First, the definition of the IANA registry for SFrame ciphersuites omits several important fields. This document updates the IANA registry specified by [RFC9605] and requests that IANA add those fields, defining the contents of those fields for current entries. Second, the AEAD construction based on AES-CTR and HMAC is defined only for the 128-bit security level. This document registers parallel constructions at the 256-bit security level.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://bifurcation.github.io/sframe-iana-256/draft-barnes-sframe-iana-256.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-barnes-sframe-iana-256/.

Discussion of this document takes place on the Secure Media Frames Working Group mailing list (mailto:sframe@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/sframe/. Subscribe at https://www.ietf.org/mailman/listinfo/sframe/.

Source for this draft and an issue tracker can be found at https://github.com/bifurcation/sframe-iana-256.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

This Internet-Draft will expire on 29 October 2026.

Copyright Notice

Table of Contents

1. Introduction 3
2. AES-256-CTR with HMAC-SHA512 3
3. Security Considerations 3
4. IANA Considerations 4
  4.1. "SFrame Cipher Suites" Registry Update 4
  4.2. Cipher Suites for AES-256-CTR with HMAC-SHA512 5
5. References 5
  5.1. Normative References 5
  5.2. Informative References 6
Appendix A. Test Vectors 6
  A.1. AEAD Encryption/Decryption Using AES-CTR and HMAC 6
  A.2. SFrame Encryption/Decryption 7
Authors' Addresses 10

1. Introduction

The Secure Frames (SFrame) protocol specification defines an IANA registry for cipher suites. The initial definition in Section 8.1 of [RFC9605] is missing several important fields. This document updates the IANA registry specified by [RFC9605] and requests that IANA add those fields and defines the contents of those fields for current entries. We also define new entries that extend the SFrame CTR+HMAC construction to support AES-256.

This document addresses two omissions in the Secure Frames (SFrame) protocol specification [RFC9605]. First, the definition in Section 8.1 of [RFC9605] of the IANA registry for SFrame ciphersuites omits several important fields. This document requests that IANA add those fields and defines the contents of those fields for current entries. Second, the AEAD construction based on AES-CTR and HMAC (in Section 4.5.1 of [RFC9605]) is defined only for the 128-bit security level. This document registers parallel constructions at the 256-bit security level.

2. AES-256-CTR with HMAC-SHA512

Section 4.5.1 of [RFC9605] defines a compound authenticated encryption construction, using the unauthenticated CTR mode of AES for encryption and HMAC for authentication.

The original specification only defines cipher suite values for instances of this construction using AES-128-CTR and HMAC-SHA256. The construction works the same way when used with AES-256-CTR and HMAC-SHA512. The only differences are in the lengths of some SFrame-internal fields:

* The keys generated by SFrame-internal key derivation (derive_key_salt) are longer to match the needs of AES-256-CTR and HMAC-SHA512 (96 bytes vs 48 bytes for AES-128-CTR and HMAC-SHA256).

* The initial tag value tag in compute_tag is 64 bytes instead of 32 bytes.

Identifiers for cipher suites using AES-256-CTR and HMAC-SHA512 are defined in Section 4.2.

3. Security Considerations

The registry changes in this document have no affect on the security of SFrame.

The new algorithms registered by this document allow the CTR+HMAC construction to be used in environments that require a 256-bit security level.

4. IANA Considerations

This document makes three requests of IANA: Updating the columns in the "SFrame Cipher Suites" registry, adding entries to the updated registry for the new cipher suites defined in this document, and add this document as an additional reference for this registry.

4.1. "SFrame Cipher Suites" Registry Update

The SFrame Cipher Suites registry should be updated to add the following columns:

* Nh: The size in bytes of the output of the hash function
* Nka: For cipher suites using the compound AEAD described in Section 4.5.1 of [RFC9605], the size in bytes of a key for the underlying encryption algorithm
* Nk: The size in bytes of a key for the encryption algorithm
* Nn: The size in bytes of a nonce for the encryption algorithm
* Nt: The overhead in bytes of the encryption algorithm (typically the size of a "tag" that is added to the plaintext)

Table 1 illustrates the new structure of the registry, and provides the required values for the currently registered entries.

Value	Name	Nh	Nka	Nk	Nn	Nt	R	Reference	Change Controller
0x0000	Reserved	-	-	-	-	-	-	RFC 9605	IETF
0x0001	AES_128_CTR_HMAC_SHA256_80	32	16	48	12	10	Y	RFC 9605	IETF
0x0002	AES_128_CTR_HMAC_SHA256_64	32	16	48	12	8	Y	RFC 9605	IETF
0x0003	AES_128_CTR_HMAC_SHA256_32	32	16	48	12	4	Y	RFC 9605	IETF
0x0004	AES_128_GCM_SHA256_128	32	n/a	16	12	16	Y	RFC 9605	IETF
0x0005	AES_256_GCM_SHA512_128	64	n/a	32	12	16	Y	RFC 9605	IETF
0xF000 - 0xFFFF	Reserved for Private Use	-	-	-	-	-	-	RFC 9605	IETF

Table 1: New structure and contents of the SFrame Cipher Suites registry

4.2. Cipher Suites for AES-256-CTR with HMAC-SHA512

The following new entries should be added to the SFrame Cipher Suites registry:

Value	Name	Nh	Nka	Nk	Nn	Nt	R	Reference	Change Controller
0x0006	AES_256_CTR_HMAC_SHA512_80	64	32	96	12	10	Y	RFC XXXX	IETF
0x0007	AES_256_CTR_HMAC_SHA512_64	64	32	96	12	8	Y	RFC XXXX	IETF
0x0008	AES_256_CTR_HMAC_SHA512_32	64	32	96	12	4	Y	RFC XXXX	IETF

Table 2: New entries SFrame Cipher Suites registry

5. References

5.1. Normative References

[RFC9605] Omara, E., Uberti, J., Murillo, S. G., Barnes, R., Ed., and Y. Fablet, "Secure Frame (SFrame): Lightweight Authenticated Encryption for Real-Time Media", RFC 9605, DOI 10.17487/RFC9605, August 2024, <https://www.rfc-editor.org/rfc/rfc9605>.

5.2. Informative References

[TestVectors] "SFrame Test Vectors", September 2025, <https://github.com/bifurcation/sframe-iana-256/blob/main/test-vectors/test-vectors-aes256.json>.

Appendix A. Test Vectors

This section provides a set of test vectors that implementations can use to verify that they correctly implement SFrame encryption and decryption with the cipher suites registered in this document. Test vectors are provided for both the AES-256-CTR-HMAC construction and for full SFrame encryption with the new cipher suites.

All values are either numeric or byte strings. Numeric values are represented as hex values, prefixed with 0x. Byte strings are represented in hex encoding.

Line breaks and whitespace within values are inserted to conform to the width requirements of the RFC format. They should be removed before use.

These test vectors are also available in JSON format at [TestVectors]. In the JSON test vectors, numeric values are JSON numbers and byte string values are JSON strings containing the hex encoding of the byte strings.

A.1. AEAD Encryption/Decryption Using AES-CTR and HMAC

For each case, we provide:

* cipher_suite: The index of the cipher suite in use (see Section 4.1)

* key: The key input to encryption/decryption

* enc_key: The encryption subkey produced by the derive_subkeys() algorithm

* auth_key: The authentication subkey produced by the derive_subkeys() algorithm

* nonce: The nonce input to encryption/decryption

* aad: The aad input to encryption/decryption

* pt: The plaintext

* ct: The ciphertext

An implementation should verify that the following are true, where AEAD.Encrypt and AEAD.Decrypt are as defined in Section 4.5.1 of [RFC9605]:

* AEAD.Encrypt(key, nonce, aad, pt) == ct

* AEAD.Decrypt(key, nonce, aad, ct) == pt

The other values in the test vector are intermediate values provided to facilitate debugging of test failures.

A.2. SFrame Encryption/Decryption

For each case, we provide:

* cipher_suite: The index of the cipher suite in use (see Section 4.1)

* kid: A KID value

* ctr: A CTR value

* base_key: The base_key input to the derive_key_salt algorithm

* sframe_key_label: The label used to derive sframe_key in the derive_key_salt algorithm

* sframe_salt_label: The label used to derive sframe_salt in the derive_key_salt algorithm

* sframe_secret: The sframe_secret variable in the derive_key_salt algorithm

* sframe_key: The sframe_key value produced by the derive_key_salt algorithm

* sframe_salt: The sframe_salt value produced by the derive_key_salt algorithm

* metadata: The metadata input to the SFrame encrypt algorithm

* pt: The plaintext

* ct: The SFrame ciphertext

An implementation should verify that the following are true, where encrypt and decrypt are as defined in Section 4.4 of [RFC9605], using an SFrame context initialized with base_key assigned to kid:

* encrypt(ctr, kid, metadata, plaintext) == ct

* decrypt(metadata, ct) == pt

The other values in the test vector are intermediate values provided to facilitate debugging of test failures.

```
cipher_suite: 0x0006
kid: 0x0000000000000123
ctr: 0x0000000000004567
base_key: 000102030405060708090a0b0c0d0e0f
sframe_key_label: 534672616d6520312e30205365637265
                  74206b6579200000000000001230006
sframe_salt_label: 534672616d6520312e30205365637265
                   742073616c74200000000000000012300
                   06
sframe_secret: 0fc3ea6de6aac97a35f194cf9bed94d4
               b5230f1cb45a785c9fe5dce9c188938a
               b6ba005bc4c0a19181599e9d1bcf7b74
               aca48b60bf5e254e546d809313e083a3
sframe_key: 3c343886ec1c79278836863e00fe934c
            8894460cfa367ebdc4856b0a9268a4f4
            fb99437876819394ef90b10ee12602d0
            23f7128ee50f2314c2cc3cff4c56616d
            2fe03ad2a254cc2ed29b2a4d3f2534c0
            dda9e7c391ad1917ea07aa221dd4b224
sframe_salt: e082f7ce012ad30c87c49e3f
metadata: 4945544620534672616d65205747
nonce: e082f7ce012ad30c87c4db58
aad: 990123456749455446205346726l6d65
     205747
pt: 64726166742d696574662d736672616d
    652d656e63
ct: 9901234567b369e03ec6467ad505ddc8
    4914115069280c5c797555be6e32cde6
    ac25bc9e
```

```
   cipher_suite: 0x0007
   kid: 0x0000000000000123
   ctr: 0x0000000000004567
   base_key: 000102030405060708090a0b0c0d0e0f
   sframe_key_label: 534672616d6520312e30205365637265
                     74206b6579200000000000000001230007
   sframe_salt_label: 534672616d6520312e30205365637265
                      742073616c74200000000000000012300
                      07
   sframe_secret: 0fc3ea6de6aac97a35f194cf9bed94d4
                  b5230f1cb45a785c9fe5dce9c188938a
                  b6ba005bc4c0a19181599e9d1bcf7b74
                  aca48b60bf5e254e546d809313e083a3
   sframe_key: 7271d6c6cbccd2e2343d480ebea65718
               a7bb379eefcf3f8d107c1e2a76e75529
               3a497fd9e4e8291b965161987ef4ef24
               983eabb06cb0a392defaab18654780a3
               9c106ffa4a47d4183a6e593cd0c1bcab
               2b9c6dcf049215845bfb7580c4dea80e
   sframe_salt: 46b4367993a314910d4d9f3d
   metadata: 4945544620534672616d65205747
   nonce: 46b4367993a314910d4dda5a
   aad: 990123456749455446205346726l6d65
        205747
   pt: 64726166742d696574662d736672616d
       652d656e63
   ct: 990123456797cb5644d8831ff8bdc080
       249990b24b569144cab2a87be22c20d9
       7976
```

```
   cipher_suite: 0x0008
   kid: 0x0000000000000123
   ctr: 0x0000000000004567
   base_key: 000102030405060708090a0b0c0d0e0f
   sframe_key_label: 534672616d6520312e30205365637265
                     74206b6579200000000000000001230008
   sframe_salt_label: 534672616d6520312e30205365637265
                      742073616c74200000000000000012300
                      08
   sframe_secret: 0fc3ea6de6aac97a35f194cf9bed94d4
                  b5230f1cb45a785c9fe5dce9c188938a
                  b6ba005bc4c0a19181599e9d1bcf7b74
                  aca48b60bf5e254e546d809313e083a3
   sframe_key: afe92c81e0df8c00fab619e0559fe5ae
               efce1ef77789d4c728af1b1c1f2e3552
               c405d274415a5291ec075c2d9954c450
               fbd36682a4e978494808b703ce78b409
               f9fec29b91e6e703a75c4131377c80c9
               d51b8906088092452e2593eb142eea2d
   sframe_salt: f6de647bac1263524cfb6533
   metadata: 4945544620534672616d65205747
   nonce: f6de647bac1263524cfb2054
   aad: 9901234567494554462053467261 6d65
        205747
   pt: 64726166742d696574662d736672616d
       652d656e63
   ct: 9901234567112a94a288b85b49ffef1d
       279f2830165c39d76cac8884011c
```

Authors' Addresses

Richard Barnes
Cisco
Email: rlb@ipv.sx

Emad Omara
Apple
Email: eomara@apple.com

Aron Rosenberg
Apple
Email: aron.rosenberg@apple.com