Short Cut Resistant AEAD Mode
Draft

Shay Gueron[1,2], Colm MacCárthaigh[2], and Alex Weibel[2]

[1]University of Haifa, Israel [2]Amazon Web Services, USA

Version of: Monday 9th December, 2019 (@ 19:23 UTC)

Abstract. Short Cut Resistant AEAD Mode (SCRAM) is a new AEAD encryption mode that addresses the tempting short cut that a decrypting party may apply: "Releasing Unauthenticated Plaintext" (RUP). SCRAM is a probabilistic encryption scheme. It uses a random value R, applies a key derivation that separates the encryption and authentication keys, computes an intermediate authentication tag T, and encrypts R with a key that depends on T. The authentication tag Tag is derived from T and some other metadata. This forces the decryption to start with authenticating the AAD and the ciphertext to retrieve T, and only then start with the actual decryption. The overhead introduced by SCRAM, compared to AES-GCM, is minimal. In addition, SCRAM has a built-in mechanism that helps obscuring traffic analysis: encryption receives an input frame size f and (if needed) pads the plaintext to the next boundary of this frame, so that the ciphertext's length is divisible by f.

1 Overview of SCRAM (simple option with no built-in message padding)

Let $\Pi = (Gen; Enc; Dec)$ be a nonce base Authenticated Encryption with Associated Data (AEAD) scheme, where encryption takes a nonce (N), a header (A), a message (M), and outputs ciphertext C plus an authentication tag T. The decrypting party receives (N, A, C, T) and invoked the Dec procedure. It is supposed to *first* authenticate the input (i.e., verify T) and *only then* conditionally release the decrypted message to the its environment. The RUP scenario is the case where the decrypting party is motivated to apply a performance short cut: trickle out the decrypted message while authenticating it on-the-fly, speculatively start some follow-up processing on the data, and determine the authenticity in the end. Here, the hope is that the consequences of the speculative data release can be undone if the message is eventually declared unauthentic. However, security cannot rely on such hopes. Some prominent AEAD modes such as AES-GCM facilitate the RUP short cut, and unfortunately the sender of an encrypted message cannot prevent the intended receiver from taking it.

1.1 Notation

A byte is an element of $\{0, 1, \ldots, 255\}$ (typically encoded by 8 bits). We use $\{0, 1\}^s$ to denote the set of all the strings of s bits, $\{0, 1\}^{8s}$ to denote the set of all the strings of

s bytes. Specific byte values are written with a $\mathsf{0x}$ prefix (e.g., $\mathsf{0x04}$ for the byte "4"). Repeated zero bytes are denoted by $\mathsf{0x00}^{\cdot}$, e.g., $\mathsf{0x00}^3$ is the string $\mathsf{0x00} \parallel \mathsf{0x00} \parallel \mathsf{0x00}$ (equivalently, $\mathsf{0x000000}$) of length 3 bytes.

With no loss of generality, we assume that inputs, outputs and values are strings of bytes. For a value V the length (in bytes) of V is denoted by $|V|$ or by $len(V)$. For short, we omit the "bytes" when specifying lengths, e.g., we say that the string $J = \mathsf{0x04} \parallel \mathsf{0x03} \parallel \mathsf{0x02} \parallel \mathsf{0x01} \parallel \mathsf{0x00}$ (equivalently $\mathsf{0x0403020100}$) has length 5. We adopt a convention where the bytes of V are listed as $v_{|V|-1}v_{|V|-1} \dots v_0$ and v_0 is considered as the "least significant byte". For example the least significant byte of $J = \mathsf{0x04} \parallel \mathsf{0x03} \parallel \mathsf{0x02} \parallel \mathsf{0x01} \parallel \mathsf{0x00}$ is $\mathsf{0x00}$ (and the most significant byte is $\mathsf{0x04}$). For $V = v_{|V|-1}v_{|V|-1} \dots v_0$ and $U = u_{|U|-1}u_{|U|-1} \dots u_0$ the concatenation of U and V is denoted by $U \parallel V$ and refers to the string $u_{|U|-1}u_{|U|-1} \dots u_0 v_{|V|-1}v_{|V|-1} \dots v_0$ of length $|U| + |V|$. For α, β such that $0 \leq \alpha < \beta \leq |V| - 1$, we use $V[\beta : \alpha]$ to denote the sub-string of $(\beta - \alpha + 1)$ bytes $v_\beta v_{\beta-1} \dots v_\alpha$. For example, $V[1 : 0] = v_1 v_0$.

Consider an integer $0 < a \leq 2^{8w} - 1$ and $w = \lceil (\log_2(a)/8) \rceil$, so that a can be written in base 16 with at most w digits. For an integer $\ell \geq w$, we use $\mathsf{str}_\ell(a)$ to denote the ℓ-byte string representation of a written in base 16 (with leading zero bytes as needed). We call it the encoding of a as a string of ℓ bytes. For example, for $a = 12825249$ (in base 10) and $w = 3$, the encoding of a as a string of 3 bytes is $\mathsf{str}_3(a) = \mathsf{0xc3b2a1}$, and the encoding of a as a string of 8 bytes is $\mathsf{str}_8(a) = \mathsf{0x0000000000c3b2a1}$.

Given the encoding $c = \mathsf{str}_\ell(a)$ of an integer a, the inverse operation that retrieves a is denoted by $\mathsf{int}(c)$. For example, $\mathsf{int}(\mathsf{0x07e3}) = 2019$.

Uniform random sampling of an element w from a finite set W is denoted by $w \leftarrow_\$ W$.

2 SCRAM with padding to the next boundary of a frame

We build an AEAD scheme SCRAM that consists of the three algorithms Gen_{SCRAM}, Enc_{SCRAM}, Dec_{SCRAM}, and is parametrized by the lengths (and limits) of the involved operands as follows. Non-negative parameters A_{max}, M_{max}, f, fbytes, r, b, nlen, t, s, u, s, w, κ, κ_1. These satisfy $\kappa_1 \geq \kappa \geq r \geq \mathsf{t} > 0$, $\mathsf{s} = \mathsf{nlen} + \mathsf{b} + 2\mathsf{u} + \mathsf{t} + r$, $0 \leq \mathsf{f} < 2^{8\mathsf{fbytes}}$, $\mathsf{w} \geq \max(\kappa, r) + \mathsf{fbytes}$. The scheme's key is denoted by K where $|K| = \kappa_1$ (bytes).

SCRAM is a composition of a privacy-only encryption scheme $\mathsf{ENCRYPT}/\mathsf{DECRYPT}$, a message authentication scheme MAC, and a pseudorandom function family F.

- The pseudorandom function family. The family $F : \{0,1\}^{8\kappa_1} \times \{0,1\}^{8\mathsf{s}} \rightarrow \{0,1\}^{8\mathsf{w}}$ is indexed by the key K, operates on an input string S of s bytes, and delivers a w bytes output. Its operation is denoted by $F(K, S)$.
- The nonce-based encryption scheme $\mathsf{ENCRYPT}/\mathsf{DECRYPT}$. We use the standard notation $C = \mathsf{ENCRYPT}(K_e, N, M)$ for encryption of the message M with the nonce N under the key K_e ($|K_e| = \kappa$) to produce ciphertext C, and $M = \mathsf{DECRYPT}(K_e, N, C)$ for decryption. The nonce length is nlen, $0 \leq |M| \leq M_{max}$ and $|C| = |M|$.
- The nonce-based message authentication scheme MAC. We use the standard notation $\mathsf{T} = \mathsf{MAC}(K_M, N, P)$ for tagging a message P with the nonce N, under the key K_M ($|K_M| = \kappa$) , to produce the tag T and $(pass/fail) = \mathsf{VER}(K_M, N, P, T)$ to denote verification. The nonce length is nlen, $0 \leq |P| \leq A_{max} + M_{max}$, and $|\mathsf{T}| = \mathsf{t}$.

Gen_{SCRAM} selects K from $\{0,1\}^{8\kappa_1}$, uniformly at random. The input to Enc_{SCRAM} is a triple (N, A, M) (nonce, Additional Authenticated Data, message) and and a frame size f, where $|N| = \mathsf{nlen}$, $0 \leq |A| \leq A_{max}$, $0 \leq |M| \leq M_{max}$, $0 \leq \mathsf{f} \leq 2^{8\mathsf{fbytes}}$. Before encryption, the message M is padded (if needed; to the next boundary of the f) into a padded message denoted $\mathsf{padded}M$, using a string of padlen zero byte, such that $|\mathsf{padded}M|$ is a multiple of f. The output is ciphretext C ($|C| = |\mathsf{padded}M|$), a value X with $|X| = r + \mathsf{fbytes}$, and an authentication tag Tag with $|Tag| = \mathsf{t}$. The flow includes the generation of a random value R of length r (X is the encryption of $R \,\|\, \mathsf{str}_{\mathsf{fbytes}}(\mathsf{padlen})$) and formatting four strings $S1$, $S2$, $S3$, $S4$ of length s ($= \mathsf{nlen} + \mathsf{b} + 2\mathsf{u} + \mathsf{t} + r$) that are input to F.

```
SCRAM encryption ($Enc_{\mathsf{SCRAM}}$)
Input: $(N, A, M)$, $\mathsf{f}$, $\mathsf{fbytes}$
Output: $C, X, Tag$
Key: $K$
  $R \longleftarrow_{\$} \{0,1\}^{8r}$
  $S1 = N \,\|\, \mathsf{0x00}^{\mathsf{b}-1} \,\|\, \mathsf{0x01} \,\|\, \mathsf{0x00}^{\mathsf{u}} \,\|\, \mathsf{0x00}^{\mathsf{u}} \,\|\, \mathsf{0x00}^{\mathsf{t}} \,\|\, R$
  $U1 = F(K, S1)$
  $K_e = U1[\kappa - 1 : 0]$
  $\mathsf{padlen} = 0$; if $\mathsf{f} > 0$ then $\mathsf{padlen} = (\mathsf{f} - len(M)) \pmod{\mathsf{f}}$
  $\mathsf{PADDING} = \mathsf{0x00}^{\mathsf{padlen}}$
  $\mathsf{padded}M = M \,\|\, \mathsf{PADDING}$
  $C = \mathsf{ENCRYPT}(K_e, N, \mathsf{padded}M)$
  $S2 = N \,\|\, \mathsf{0x00}^{\mathsf{b}-1} \,\|\, \mathsf{0x02} \,\|\, \mathsf{0x00}^{\mathsf{u}} \,\|\, \mathsf{0x00}^{\mathsf{u}} \,\|\, \mathsf{0x00}^{\mathsf{t}} \,\|\, \mathsf{0x00}^{r}$
  $U2 = F(K, S2)$
  $K_M = U2[\kappa - 1 : 0]$
  $T = \mathsf{MAC}(K_M, N, A \| C)$
  $S3 = N \,\|\, \mathsf{0x00}^{\mathsf{b}-1} \,\|\, \mathsf{0x03} \,\|\, \mathsf{0x00}^{\mathsf{u}} \,\|\, \mathsf{0x00}^{\mathsf{u}} \,\|\, T \,\|\, \mathsf{0x00}^{r}$
  $U3 = F(K, S3)$
  $Y1 = U3[r - 1 : 0] \oplus R$
  $Y0 = U3[r + \mathsf{fbytes} - 1 : r] \oplus \mathsf{str}_{\mathsf{fbytes}}(\mathsf{padlen})$
  $X = Y1 \,\|\, Y0$
  $S4 = N \,\|\, \mathsf{0x00}^{\mathsf{b}-1} \,\|\, \mathsf{0x04} \,\|\, \mathsf{str}_{\mathsf{u}}(len(A)) \,\|\, \mathsf{str}_{\mathsf{u}}(len(M)) \,\|\, T \,\|\, R$
  $U4 = F(K, S4)$
  $Tag = U4[\mathsf{t} - 1 : 0]$
  Output $C, X, Tag$
```

The input for Dec_{SCRAM} decryption is the quadruple (N, A, C, X, Tag) where $0 \leq |A| \leq A_{max}$, $0 \leq |C| \leq M_{max} + \mathsf{fbytes}$, $|X| = r + \mathsf{fbytes}$, $|Tag| = \mathsf{t}$. If the authentication passes, the output is either the decrypted message M, where $|M| = |C| - \mathsf{padlen}$, and padlen that is extracted from X. If the authentication fails, the output is a failure symbol $\perp$.

```
SCRAM decryption ($Dec_{\mathsf{SCRAM}}$)
Input: $(N, A, C, X, Tag)$, $\mathsf{fbytes}$
Output: either $M$ or $\perp$
Key: $K$
  $S2 = N \,\|\, \mathsf{0x00}^{\mathsf{b}-1} \,\|\, \mathsf{0x02} \,\|\, \mathsf{0x00}^{\mathsf{u}} \,\|\, \mathsf{0x00}^{\mathsf{u}} \,\|\, \mathsf{0x00}^{\mathsf{t}} \,\|\, \mathsf{0x00}^{r}$
  $U2 = F(K, S2)$
  $K_M = U2[\kappa - 1 : 0]$
  $T = \mathsf{MAC}(K_M, N, A \| C)$
```

```
$S3 = N \parallel \mathsf{0x00}^{\mathsf{b}-1} \parallel \mathsf{0x03} \parallel \mathsf{0x00}^{\mathsf{u}} \parallel \mathsf{0x00}^{\mathsf{u}} \parallel T \parallel \mathsf{0x00}^{r}$
$U3 = F(K, S3)$
$Y1 = U3[r-1:0] \oplus X[r-1:0]$
$R = Y1$
$Y0 = U3[r+\mathsf{fbytes}-1:r] \oplus X[r+\mathsf{fbytes}-1:r]$
$\mathsf{padlen} = \mathsf{int}_{\mathsf{fbytes}}(Y0)$
$LM = len(C) - \mathsf{padlen}$
$S4 = N \parallel \mathsf{0x00}^{\mathsf{b}-1} \parallel \mathsf{0x04} \parallel \mathsf{str}_{\mathsf{u}}(len(A)) \parallel \mathsf{str}_{\mathsf{u}}(LM) \parallel T \parallel R$
$U4 = F(K, S4)$
$Tag' = U4[\mathsf{t}-1:0]$
if $Tag' \neq Tag$
    Output $\perp$
else
    $S1 = N \parallel \mathsf{0x00}^{\mathsf{b}-1} \parallel \mathsf{0x01} \parallel \mathsf{0x00}^{\mathsf{u}} \parallel \mathsf{0x00}^{\mathsf{u}} \parallel \mathsf{0x00}^{\mathsf{t}} \parallel R$
    $U1 = F(K, S1)$
    $K_e = U1[\kappa-1:0]$
    $\mathsf{padded}M = \mathsf{DECRYPT}(K_e, N, C)$
    $M = \mathsf{padded}M[len(C)-1:\mathsf{padlen}]$
    Output $M$
```

A specific instantiation of SCRAM. In a specific instantiation, we set $M_{max} = A_{max} = 2^{32}$, $\mathsf{nlen} = 12$, $\mathsf{t} = 16$, $\mathsf{b} = 4$, $\kappa_1 = \kappa = r = 32$, $\mathsf{fbytes} = 2$, $\mathsf{u} = 8$, $\mathsf{s} = 80$, $\mathsf{w} = 64$.

This implies that the inputs to F have length $\mathsf{s} = \mathsf{nlen} + \mathsf{b} + 2\mathsf{u} + \mathsf{t} + r = 80$, that $\mathsf{f} < 2^{16}$, so the size of the padding can be encoded in 2 bytes, and that For ENCRYPT/DECRYPT, we use Counter Mode with the block cipher AES. For MAC we use GMAC (with a 32-byte key). For F we use HMAC-SHA512 (with digest size 64).