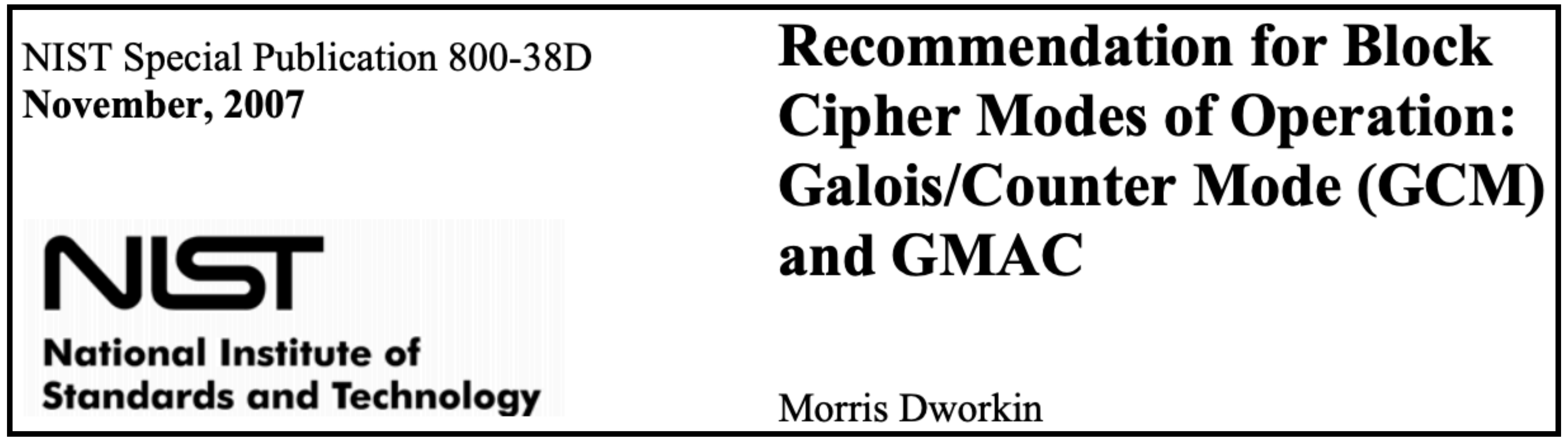
V5b
AES-GCM

AUTHENTICATED ENCRYPTION

CRYPTO 101: Building Blocks

cryptography101.ca

Overview


NIST Special Publication 800-38D
November, 2007

NIST
National Institute of
Standards and Technology

Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC

Morris Dworkin

- AES-GCM is an authenticated encryption scheme designed by David McGrew and John Viega in 2004.
- Adopted as a NIST standard (SP 800-38D) in 2007.
- Uses the CTR mode of encryption and GMAC, a custom-designed MAC scheme.

CTR: CounTeR mode of encryption

Let $k \in_R \{0,1\}^{128}$ be the secret key shared by Alice and Bob. Let $M = (M_1, M_2, \ldots, M_u)$ be a plaintext message, where each M_i is a 128-bit block and $u \le 2^{32} - 2$.

To encrypt M, Alice does the following:

1. Select a nonce $IV \in \{0,1\}^{96}$.
2. Let $J_0 = IV \| 0^{31} \| 1$.
3. For i from 1 to u do:
   $J_i \leftarrow J_{i-1} + 1$ and compute
   $C_i = \mathrm{AES}_k(J_i) \oplus M_i$.
4. Send $(IV, C_1, C_2, \ldots, C_u)$ to Bob.

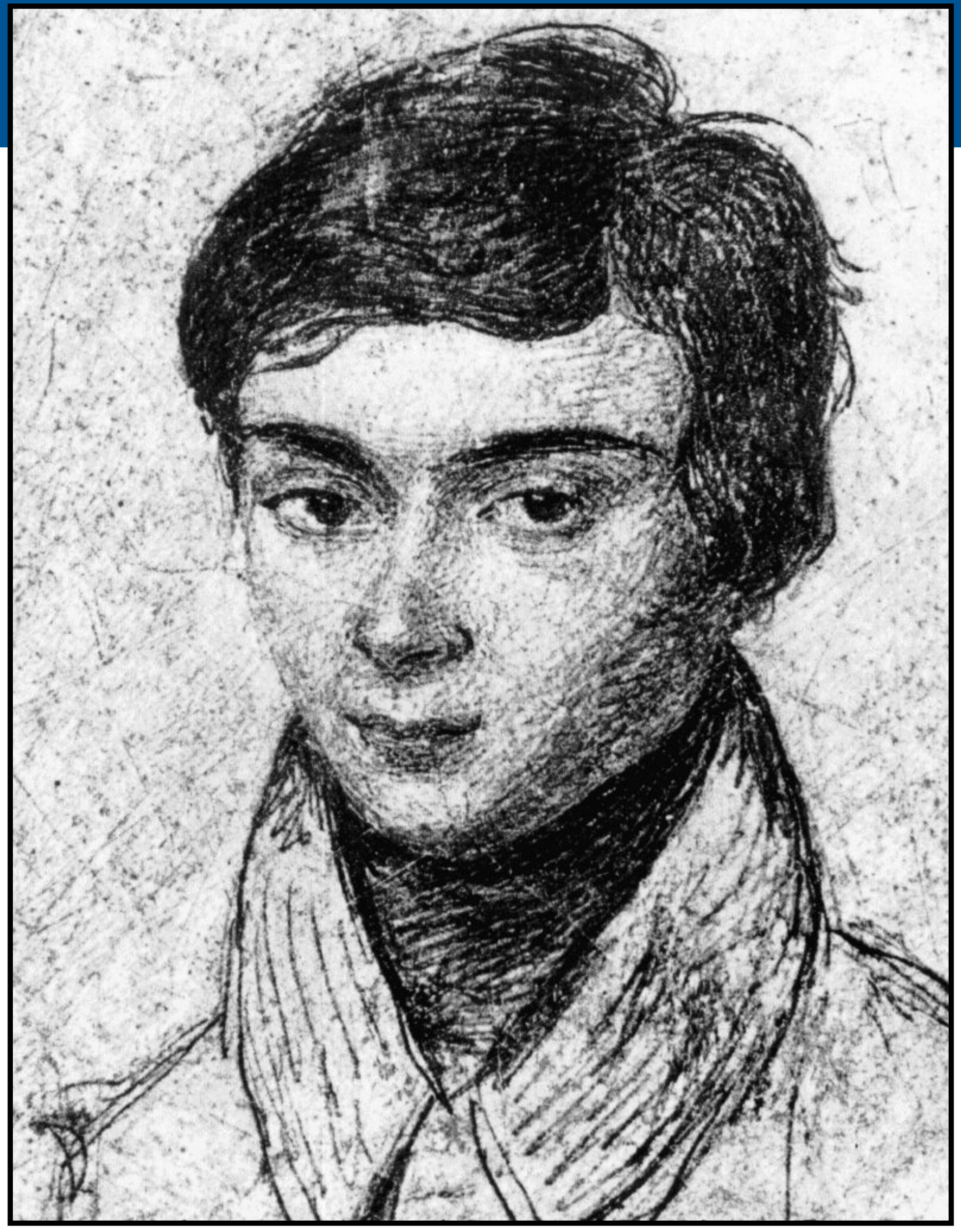
To decrypt, Bob does the following:

1. Let $J_0 = IV \| 0^{31} \| 1$.
2. For i from 1 to u do:
   $J_i \leftarrow J_{i-1} + 1$ and compute
   $M_i = \mathrm{AES}_k(J_i) \oplus C_i$.

Notes on CTR mode

1. CTR mode of encryption can be viewed as a stream cipher.
2. As was the case with CBC encryption, identical plaintexts with different IVs result in different ciphertexts.
3. It is critical that the IV should not be repeated, but this can be difficult to achieve in practice.
4. Unlike CBC encryption, CTR encryption is parallelizable.
5. Note that AES^{-1} is not used.
6. The secret key can have bitlength 128, 192 or 256.

Multiplying blocks



Évariste Galois

- Let $a = a_0a_1a_2 \ldots a_{127}$ be a 128-bit block.
  We associate the binary polynomial
  $a(x) = a_0 + a_1x + a_2x^2 + \cdots + a_{127}x^{127} \in \mathbb{Z}_2[x]$ with a.
- Let $f(x) = 1 + x + x^2 + x^7 + x^{128}$.
- If a and b are 128-bit blocks, then define $c = a \bullet b$ to be the block corresponding to the polynomial $c(x) = a(x) \cdot b(x) \bmod f(x)$.
  - That is, $c(x)$ is the remainder upon dividing $a(x) \cdot b(x)$ by $f(x)$ in $\mathbb{Z}_2[x]$.
  - This is multiplication in the Galois field $GF(2^{128})$.

Galois Message Authentication Code (GMAC)

- Let $A = (A_1, A_2, \ldots, A_v)$, where each A_i is a 128-bit block.
- Let L be the bitlength of A (encoded as a 128-bit block).
- Let $k \in_R \{0,1\}^{128}$ be the secret key.

1. Let $J_0 = IV \| 0^{31} \| 1$, where $IV \in \{0,1\}^{96}$ is a nonce.
2. Compute $H = \mathrm{AES}_k(0^{128})$.
3. Let $f_A(x) = A_1 x^{v+1} + A_2 x^v + \cdots + A_{v-1} x^3 + A_v x^2 + Lx \in GF(2^{128})[x]$.
4. Compute the authentication tag $t = \mathrm{AES}_k(J_0) \oplus f_A(H)$.
5. Send (IV, A, t).

Computing $f_A(H)$ using Horner's rule

- Example: Let $A = (A_1, A_2, A_3)$.
  - Then $f_A(x) = A_1x^4 + A_2x^3 + A_3x^2 + Lx$.
  - Hence, $f_A(H) = A_1H^4 + A_2H^3 + A_3H^2 + LH$.
  - $f_A(H)$ can be computed using Horner's rule:
    $$f_A(H) = ((((((A_1 \cdot H) + A_2) \cdot H) + A_3) \cdot H) + L) \cdot H.$$
  - This requires three additions and four multiplications in $GF(2^{128})$.
- In general, if A has blocklength v, then computing $f_A(H)$ using Horner's rule requires v additions and $v + 1$ multiplications in $GF(2^{128})$.

Security argument

- Consider the simplified tag: $t' = f_A(H)$.
  - An adversary can guess the tag t' of a message A with success probability $\frac{1}{2^{128}}$.
  - She can also guess the tag t' by making a guess H' for H and computing $f_A(H')$. Her success probability is at most $\frac{v+1}{2^{128}}$, where v is the blocklength of A .
  - However, if the adversary sees a single valid message-tag pair (A, t'), she can solve the polynomial equation $f_A(H) = t'$ for H.
- To circumvent the aforementioned attack, a second secret $\mathrm{AES}_k(J_0)$ is used to hide t': $t = \mathrm{AES}_k(J_0) \oplus f_A(H)$. The secret $\mathrm{AES}_k(J_0)$ serves as a one-time pad for t'.

Authenticated encryption: AES-GCM

Input:

- AAD (Additional Authenticated Data), also called encryption context: Data to be authenticated (but not encrypted): $A = (A_1, A_2, \ldots, A_v)$.
- Data to be encrypted and authenticated: $M = (M_1, M_2, \ldots, M_u)$, $u \le 2^{32} - 2$.
- Secret key: $k \in_R \{0,1\}^{128}$, shared between Alice and Bob

Output: (IV, A, C, t), where

- IV is a 96-bit initialization vector.
- $A = (A_1, A_2, \ldots, A_v)$ is the additional authenticated data.
- $C = (C_1, C_2, \ldots, C_u)$ is the encrypted/authenticated data.
- t is a 128-bit authentication tag.

AES-GCM encryption/authentication

Alice does the following:

1. Let $L = L_A \| L_M$, where L_A, L_M are the bitlengths of A, M expressed as 64-bit integers. (L is the length block.)
2. Select a nonce $IV \in \{0,1\}^{96}$ and let $J_0 = IV \| 0^{31} \| 1$.
3. Encryption: For i from 1 to u do:
   Compute $J_i = J_{i-1} + 1$ and $C_i = \mathrm{AES}_k(J_i) \oplus M_i$.
4. Authentication: Compute $H = \mathrm{AES}_k(0^{128})$.
   Compute $t = \mathrm{AES}_k(J_0) \oplus f_{A,C}(H)$.
5. Output: (IV, A, C, t).

Note: $f_{A,C}(x) = A_1 x^{u+v+1} + A_2 x^{u+v} + \cdots + A_{v-1} x^{u+3} + A_v x^{u+2} + C_1 x^{u+1} + C_2 x^u + \cdots + C_{u-1} x^3 + C_u x^2 + Lx$

AES-GCM decryption/authentication

Upon receiving (IV, A, C, t), Bob does the following:

1. Let $L = L_A \| L_C$, where L_A, L_C are the bitlengths of A, C expressed as 64-bit integers.
2. Authentication: Compute $H = \text{AES}_k(0^{128})$.
   Compute $t' = \text{AES}_k(J_0) \oplus f_{A,C}(H)$.
   If $t' = t$ then proceed to decryption; if $t' \neq t$ then reject.
3. Decryption: Let $J_0 = IV \| 0^{31} \| 1$.
   For i from 1 to u do:
   Compute $J_i = J_{i-1} + 1$ and $M_i = \text{AES}_k(J_i) \oplus C_i$.
4. Output: (A, M).

Some features of AES-GCM

1. Performs both authentication and encryption.
2. Supports authentication only (by using empty M).
3. Very fast implementations on Intel and AMD processors because of special AES-NI and PCLMUL instructions for the AES and $\bullet$ operations.
4. Encryption and decryption can be parallelized.
5. AES-GCM can be used in streaming mode.
6. The secret key can have bitlength 128, 192 or 256.
7. Security is justified by a security proof:
   - The original McGrew-Viega security proof (2004) was wrong.
   - The proof was fixed in 2012 by Iwata-Ohashi-Minematsu.

Performance

Speed benchmarks[†] from 2018 on an Intel Xeon CPU (E3-1220 V2) at 3.10 GHz in 64-bit mode.

[†] Relative speeds will likely be very different on other processors.

Source: www.bearssl.org/speed.html

Algorithm	block length	key length	digest length (bits)	speed (Mbytes/
ChaCha20	—	256	—-	323
Triple-DES	64	168	—-	21
AES-128	128	128	—-	170
AES-128-NI	128	128	—-	2426
AES-256	128	256	—-	129
AES-256-NI	128	256	—-	1830
GMAC	128	128	128	247
GMAC-PCLMUL	128	128	128	1741

IV's should not be repeated

IV's should not be repeated (with the same key k).

- Suppose an IV is reused, and an eavesdropper captures two transmissions: (IV, A_1, C_1, t_1), (IV, A_2, C_2, t_2). Suppose also that M_1 and M_2 have the same blocklengths, and that the eavesdropper knows M_1.
- Then $t_1 = \text{AES}_k(J_0) \oplus f_{A_1,C_1}(H)$ and $t_2 = \text{AES}_k(J_0) \oplus f_{A_2,C_2}(H)$, so $t_1 \oplus t_2 = f_{A_1,C_1}(H) \oplus f_{A_2,C_2}(H)$.
- This polynomial equation can be quickly solved for H, and then $\text{AES}_k(J_0) = t_1 \oplus f_{A_1,C_1}(H)$ can be computed.
- Thereafter, the adversary can properly encrypt/authenticate any plaintext (of blocklength at most that of M_1).