Nonce-Disrespecting Adversaries: Practical Forgery Attacks on GCM in TLS

Hanno Böck, Aaron Zauner, Sean Devlin, Juraj Somorovsky, Philipp Jovanovic
TLS Encryption

1. Asymmetric key exchange
   – RSA, DHE, ECDHE

2. Symmetric encryption
TLS Encryption

1. Asymmetric key exchange
   – RSA, DHE, ECDHE

2. Symmetric encryption
   – CBC/HMAC
   – RC4 (stream cipher)
   – (new: ChaCha20/Poly1305)
   – AES-GCM
CBC / HMAC

• Arbitrary padding in SSLv3

• Implicit IVs in TLS 1.0

• MAC-then-Pad-then-Encrypt

2014 Poodle

2011 BEAST

2002 Padding Oracles

Lucky microseconds

Lucky 13
TLS Encryption

1. Asymmetric key exchange
   – RSA, DHE, ECDHE

2. Symmetric encryption
   – CBC/HMAC
   – RC4 (stream cipher)
   – (new: ChaCha20/Poly1305)
   – AES-GCM
RC4

- Generates a key stream
  - Some bytes more likely to occur

  - 2001: Fluhrer, Mantin, Shamir
  - 2013: AlFardan et al.
  - 2015: Vanhoef, Piessens
  - 2015: Garman et al.

- https://www.rc4nomore.com/
- RFC 7465: Prohibiting RC4 Cipher Suites
TLS Encryption

1. Asymmetric key exchange
   – RSA, DHE, ECDHE

2. Symmetric encryption
   – CBC/HMAC
   – RC4 (stream cipher)
   – (new: ChaCha20/Poly1305)
   – AES-GCM
TLS Encryption

1. Asymmetric key exchange
   - RSA, DHE, ECDHE

2. Symmetric encryption
   - CBC/HMAC
   - RC4 (stream cipher)
   - (new: ChaCha20/Poly1305)
   - AES-GCM
Overview

1. AES-GCM
2. The Forbidden Attack
3. Evaluation
4. Attack Scenario
AES Counter Mode

Nonce || Counter

$J_1 \rightarrow AES-Enc \rightarrow P_1 \rightarrow C_1$

$J_2 \rightarrow AES-Enc \rightarrow P_2 \rightarrow C_2$
Bit Flipping in AES Counter Mode

Attacker can modify messages
AES-GCM

- GCM – Galois Counter Mode
- AEAD (Authenticated Encryption with Additional Data)
- Only in TLS 1.2
- Combination of **Counter Mode** and **GHASH** authentication
  - Computed over Galois field
AES-GCM

Hash key $H$
Encryption of 128 zero bits: $H=\text{Enc}(0)$

Output: $C \ || \ T$
GCM: Opinions of Cryptographers

• "Do not use GCM. Consider using one of the other authenticated encryption modes, such as CWC, OCB, or CCM." (Niels Ferguson)
• "We conclude that common implementations of GCM are potentially vulnerable to authentication key recovery via cache timing attacks." (Emilia Käsper, Peter Schwabe, 2009)
• "AES-GCM so easily leads to timing side-channels that I'd like to put it into Room 101." (Adam Langley, 2013)
• "The fragility of AES-GCM authentication algorithm" (Shay Gueron, Vlad Krasnov, 2013)
• "GCM is extremely fragile" (Kenny Paterson, 2015)
Overview

1. AES-GCM
2. The Forbidden Attack
3. Evaluation
4. Attack Scenario
The Forbidden Attack

• Nonce:
  – Number used once
  – TLS: 8 Byte / 64 Bit nonce

• Joux (2006): Nonce reuse allows an attacker to recover the authentication key

• Attacker can modify messages
Consider one block

\[ H = AES(0) \]

\[ T = (C_1 \cdot H + L) \cdot H + AES(J_0) \]

\[ T = C_1 \cdot H^2 + L \cdot H + AES(J_0) \]

Unknown values:

- \( H \)
- \( AES(J_0) \)
Duplicate nonce

\[ H = AES (0) \]

\[ T_1 = C_{1,1} * H^2 + L_1 * H + AES (J_0) \]
\[ T_2 = C_{2,1} * H^2 + L_2 * H + AES (J_0) \]

\[ T_1 - T_2 = (C_{1,1} - C_{2,1}) * H^2 \]
\[ + (L_1 - L_2) * H \]

Finding \( H \) possible
Overview

1. AES-GCM
2. The Forbidden Attack
3. Evaluation
4. Attack Scenario
"Each value of the nonce_explicit must be distinct for each distinct invocation of the GCM encrypt function for any fixed key. Failure to meet this uniqueness requirement can significantly degrade security. The nonce_explicit may be the 64-bit sequence number."

Two problems:
• Random nonces: Collision probability
• Repeating nonces
Internet-wide Scan

• **184** hosts with repeating nonces
  – Radware (Cavium chip)
  – Several pages from VISA Europe

• **72445** hosts with random looking nonces
  – A10, IBM Lotus Domino (both published updates)
  – Sangfor (no response)

• More devices that we were unable to identify
Example: Radware

OpenSSL 1.0.1j

e_aes.c (EVP_CIPHER_CTX_ctrl/aes_gcm_ctrl):

    if (c->encrypt &&
        RAND_bytes(gctx->iv + arg, gctx->ivlen - arg) <= 0)
        return 0;

t1_enc.c:

    if (EVP_CIPHER_mode(c) == EVP_CIPH_GCM_MODE)
    {
        EVP_CIPHERInit_ex(dd,c,NULL,key,NULL,(which & SSL3_CC_WRITE),
         EVP_CIPHER_CTX_ctrl(dd, EVP_CTRL_GCM_SET_IV_FIXED, k);
    }

0100000003001741
0100000003001741
f118cd0fa6ff5a15
f118cd0fa6ff5a16
f118cd0fa6ff5a74

No random generation return value check
Open Source Libraries

- Botan, BouncyCastle, MatrixSSL, SunJCE, OpenSSL
- No real problems
- Counter overflows in Botan and MatrixSSL
Overview

1. AES-GCM
2. The Forbidden Attack
3. Evaluation
4. Attack Scenario
Attacking Vulnerable Websites

GET visa.dk/index.html

HTTP 1.1 200 OK
...
<html>
<script> ... </script>
</html>

HTTP 1.1 200 OK
...
<html>
<h1>Hello Visa</h1>
</html>
Demo
Attacking mi5.gov.uk

HTTP/1.1 301 Moved Permanently
Strict-Transport-Security: max-age=31536000
Date: Tue, 02 Aug 2016 20:47:06 GMT
Server: Apache
X-Frame-Options: SAMEORIGIN, SAMEORIGIN
Location: https://www.mi5.gov.uk/careers?146718903ac4b72b

Cache-Control: max-age=1209600
Expires: Tue, 16 Aug 2016 20:47:06 GMT
Content-Length: 255
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>301 Moved Permanently</title>
</head><body>
<h1>Moved Permanently</h1>
<p>The document has moved <a href="https://www.mi5.gov.uk/careers?146718903ac4b72b">here</a>.</p>
</body></html>
Conclusions

• TLS 1.2: no guidance how to use nonces correctly
  – Some people get it wrong
• Implicit nonces needed:
  – Chacha20/Poly1305 and TLS 1.3 based on record number
• Better test tools for TLS implementation flaws