Revisione dei Cifrari a Blocchi: Autenticazione Cifrata (Authenticated Encryption)

Basato su "Review of Block ciphers:Authenticated Encryption"
Riferimento di studio: Aumasson, capitolo 8

Indice

1 Riepilogo: La Cifratura da sola non basta

Un concetto fondamentale è che la **cifratura non garantisce l'integrità** dei dati. Esiste una distinzione cruciale tra i due obiettivi di sicurezza:

- **Confidenzialità**: È definita come "sicurezza semantica contro CPA" (Chosen Plaintext Attack). Questo tipo di cifratura protegge solo dall'intercettazione (eavesdropping). **NON è sicura** contro attaccanti attivi, che possono modificare i dati.
- **Integrità**: È definita come "non falsificabilità sotto CCA" (Chosen Ciphertext Attack). Questo meccanismo (come un MAC) protegge dalla manomissione, ma **i messaggi rimangono in chiaro** (plaintext).

2 Authenticated Encryption (AE)

L'Autenticazione Cifrata (AE) è una modalità di cifratura progettata specificamente per essere **sicura contro attacchi attivi**, come la manomissione (tampering).

L'obiettivo dell'AE è garantire **sia la confidenzialità sia l'integrità** dei dati.

Questo è essenziale perché, come vedremo, gli attacchi di manomissione possono anche riuscire a **compromettere la confidenzialità**, non solo l'integrità.

3 Esempio 1: Attacco a CBC (Cipher Block Chaining)

Questo esempio illustra come un attaccante attivo può manipolare un ciphertext cifrato in modalità CBC per violare la confidenzialità.

3.1 Scenario

Immaginiamo un pacchetto TCP/IP cifrato (ad esempio, in un tunnel IPsec) che viene inviato a una macchina di destinazione. Il pacchetto originale è destinato alla porta 80 (WWWport). Il primo blocco di plaintext contiene l'intestazione, P_1 = "dest=80".

Un'attaccante ("Eve") intercetta questo pacchetto. L'attaccante vuole reindirizzare il pacchetto a un'altra porta, ad esempio la porta 25 (SMTP), sulla quale Eve può osservare il traffico in chiaro.

3.2 L'Attacco (Bit-Flipping sull'IV)

La decifratura del primo blocco in modalità CBC è:

$$P_1 = D_K(C_1) \oplus IV$$

L'attaccante non può modificare C_1 (il blocco cifrato) senza conoscere la chiave K. Tuttavia, può **modificare l'Initialization Vector (IV)**, che viene trasmesso in chiaro.

L'attaccante crea un nuovo IV, IV', manipolando l'originale:

$$IV' = IV \oplus (\text{"dest=80"}) \oplus (\text{"dest=25"})$$

3.3 Il Risultato

L'attaccante inoltra il pacchetto manomesso (composto da $IV', C_1, C_2, \dots$) alla macchina di destinazione. Il TCP/IP stack riceve il pacchetto e tenta di decifrarlo:

$$P_1' = D_K(C_1) \oplus IV'$$

Sostituendo IV':

$$P_1' = D_K(C_1) \oplus (IV \oplus (\text{"dest=80"}) \oplus (\text{"dest=25"}))$$

Raggruppando i termini:

$$P_1' = (D_K(C_1) \oplus IV) \oplus (\text{"dest=80"}) \oplus (\text{"dest=25"})$$

Dato che $(D_K(C_1) \oplus IV)$ è il plaintext originale P_1 (cioè "dest=80"):

$$P_1' = (\text{"dest=80"}) \oplus (\text{"dest=80"}) \oplus (\text{"dest=25"})$$

$$P_1' = \text{"dest=25"}$$

Il pacchetto viene decifrato correttamente, ma ora è destinato alla porta 25. I dati contenuti nei blocchi successivi vengono inviati a quella porta, dove l'attaccante può leggerli in chiaro.

Questo attacco dimostra che un cifrario CPA-secure (come CBC) è trivialmente vulnerabile a un Chosen Ciphertext Attack (CCA).

4 Esempio 2: Attacco CCA contro CTR (Counter Mode)

Qualcuno potrebbe pensare che il problema sia solo di CBC, e che la modalità CTR non sia vulnerabile. Questo esempio dimostra il contrario.

4.1 Scenario

Consideriamo un'applicazione di terminale remoto in cui ogni tasto premuto (1 byte di dati, D) viene cifrato e inviato usando la modalità CTR. Il pacchetto contiene anche un checksum TCP a 16 bit.

L'attaccante ha solo accesso alla rete.

L'Oracolo: Lo stack TCP/IP del destinatario elabora solo pacchetti validi. Invierà un ACK **solo se** il checksum TCP del pacchetto decifrato è corretto.

4.2 L'Attacco

In modalità CTR, il ciphertext C è $P \oplus \text{Pad}$, dove $\text{Pad} = E_K(\text{nonce})$. Il plaintext P è composto da $(cksum||D)$. L'attaccante intercetta $C = (cksum_C||D_C)$.

L'attaccante modifica C applicando uno XOR con valori scelti t (sui bit del checksum) e s (sul byte del dato):

$$C' = C \oplus (t||s)$$

Il destinatario decifra C':

$$P' = C' \oplus \text{Pad} = (C \oplus (t||s)) \oplus \text{Pad} = (P \oplus \text{Pad}) \oplus (t||s) \oplus \text{Pad} = P \oplus (t||s)$$

Quindi, i nuovi dati decifrati sono:

- $cksum' = cksum \oplus t$
- $D' = D \oplus s$

L'attaccante invia questo pacchetto modificato. Se riceve un ACK, sa che $cksum'$ è il checksum corretto per D'.

Questo crea un'equazione valida: $\text{checksum}(\text{hdr}, D \oplus s) = cksum \oplus t$. Sebbene $cksum$ e D siano sconosciuti, l'attaccante può usare questo oracolo (l'ACK) per testare ipotesi e, con un numero sufficiente di tentativi, risolvere l'equazione e recuperare il plaintext D.

5 Lezione da Imparare

La sicurezza CPA (Chosen Plaintext Attack) **non può garantire la segretezza** (confidenzialità) in presenza di attacchi attivi.

Si devono usare solo due approcci:

1. Se un messaggio richiede **solo integrità** (ma non confidenzialità), si usa un **MAC** (Message Authentication Code).
2. Se un messaggio richiede **sia integrità sia confidenzialità**, si usa una modalità di **autenticazione cifrata** (AE).

6 Definizione di Authenticated Encryption (AE)

Un algoritmo AE è una coppia di funzioni (Cifratura E e Decifratura D).

- **Cifratura**: $E(k, MSG) \to C$ (dove C include un tag di autenticazione).
- **Decifratura**: $D(k, C) \to MSG$ oppure $REJECT$.

A differenza della cifratura standard che restituisce sempre un messaggio (magari errato), un algoritmo AE può restituire un output speciale "REJECT" (Rifiutato).

L'AE è fondamentalmente più forte perché decifra un messaggio **solo se il tag di autenticazione è valido**. Questo meccanismo impedisce all'attaccante di eseguire attacchi CCA (Chosen Ciphertext Attack).

6.1 Definizione di Sicurezza: Integrità del Ciphertext

Un AE è sicuro se:

1. È semanticamente sicuro sotto CPA (garantisce la confidenzialità).
2. Garantisce l'integrità del ciphertext.

L'integrità del ciphertext viene definita (informalmente) tramite un "gioco" tra un Challenger (che possiede la chiave k) e un Avversario:

- L'Avversario invia messaggi $m_1, \ldots, m_q$ al Challenger.
- Il Challenger risponde con i ciphertext $c_1, \ldots, c_q$.
- L'Avversario produce un nuovo ciphertext c **diverso da tutti quelli ricevuti** ($c \notin \{c_1, \ldots, c_q\}$).
- **Successo dell'Avversario**: L'Avversario vince se la decifratura di c **non** produce un errore (cioè $D(k, c) \neq \perp$ o REJECT).

L'integrità è garantita se la probabilità di successo dell'avversario è trascurabile. (Nota: l'attaccante vince anche se c decifra in un messaggio casuale senza senso!).

7 AEAD: AE con Dati Associati (Associated Data)

AEAD è un'estensione dell'AE che gestisce "dati associati" (AD).

- **Dati Cifrati**: Sono protetti in **confidenzialità** e **autenticità**.
- **Dati Associati (AD)**: Sono protetti solo in **autenticità**.

I dati associati (spesso gli header dei pacchetti) devono rimanere in chiaro (plaintext) per il routing o altre elaborazioni, ma non devono essere modificabili.

L'AEAD è un concetto flessibile:

- **Nessun dato associato**: L'AEAD si riduce a una cifratura standard sicura contro CCA.
- **Nessun dato da cifrare**: L'AEAD agisce come un MAC sicuro.

Questo spiega perché i protocolli moderni, come **TLSv1.3**, usano esclusivamente AEAD. Se l'AEAD non è disponibile, l'alternativa sicura (CCA-secure) è la costruzione **Encrypt-then-MAC**.

8 Scelte Progettuali per AEAD

8.1 Struttura

- **Due livelli (Two-layer)**: Prima si cifra, poi si calcola il MAC (es. AES-GCM). Richiede spesso due passaggi sui dati (lento).
- **Un livello (One-layer)**: Cifratura e MAC "tutto in uno" (es. OCB, più veloce).

8.2 Performance

- **Parallelizzabilità**: Difficile da ottenere "pienamente", poiché il controllo di autenticità richiede (di solito) tutti i dati.
- **Streamability (Online cipher)**: Capacità di processare i dati in un unico passaggio (one pass).

8.3 Robustezza

- **Resistenza all'abuso (Misuse resistance)**: Quanto è robusto l'algoritmo se l'IV (nonce) viene riutilizzato?.

9 AES-GCM (Galois Counter Mode)

AES-GCM è stato il primo algoritmo AEAD standardizzato (NIST SP 800-38D). È ampiamente utilizzato nei protocolli di sicurezza come IPsec e TLS (ma non nel WiFi, che usa AES-CCM).

9.1 Struttura

AES-GCM utilizza una struttura **Encrypt-then-MAC**:

- **Cifratura (CM - Counter Mode)**: Viene usato **AES-CTR** (AES in modalità Counter).
- **MAC (G - Galois)**: Viene usato **GHASH**.

GHASH non è un hash crittografico (come SHA-256), ma una **funzione di hash universale** molto più veloce. Si basa sulla moltiplicazione nel campo di Galois $GF(2^{128})$ (moltiplicazione polinomiale senza riporto), spesso accelerata via hardware (es. istruzione CLMUL).

9.2 Costruzione di AES-GCM (Alto Livello)

La costruzione di GCM (mostrata con un messaggio di due blocchi, $m[1]$ e $m[2]$) avviene in due fasi:

9.2.1 1. Cifratura (AES-CTR)

I blocchi di plaintext $m[i]$ sono cifrati usando AES in modalità Counter. Un contatore viene generato partendo dall'IV (nonce):

- $ct[1] = m[1] \oplus AES_K(IV|1)$
- $ct[2] = m[2] \oplus AES_K(IV|2)$

9.2.2 2. Autenticazione (GHASH e Wegman-Carter)

Il MAC GCM è una costruzione di tipo **Wegman-Carter**.

1. **Creazione della chiave H (Kauth)**: Si genera una chiave di autenticazione H (Kauth) cifrando un blocco di zeri con la chiave K: $H = AES_K(0...0)$. H è la chiave usata da GHASH. (Le chiavi di cifratura K e di autenticazione H devono essere diverse).
2. **Hashing (GHASH)**: Si applica la funzione GHASH (indicata come "GM" o "GMH") ai blocchi del **ciphertext** $(ct[1], ct[2])$ e alla lunghezza dei dati $(Len(ct))$. (È fondamentale autenticare la lunghezza).
3. **Gestione Dati Associati (AD)**: Se presenti, i blocchi $ad[1], ad[2], \ldots$ vengono inseriti in GHASH **prima** dei blocchi di ciphertext. La lunghezza totale $Len(ad{+}ct)$ viene aggiornata per includere anche l'AD.
4. **Finalizzazione (Tag)**: L'output di GHASH (che non è crittografico) viene reso sicuro tramite uno XOR finale con un pad cifrato. Si usa il contatore $IV|0$:

$$\text{Auth tag} = (\text{GHASH}_H(\text{AD}, \text{ct}, \text{Len})) \oplus AES_K(IV|0)$$

L'inclusione dell'IV nel calcolo del tag (tramite $AES_K(IV|0)$) è cruciale per prevenire attacchi come quello visto nell'Esempio 1.

9.3 MAC Wegman-Carter

La costruzione Wegman-Carter (WC) permette di creare un MAC sicuro utilizzando funzioni di hash **non crittografiche** (che sono molto veloci). Richiede due blocchi:

1. **Universal Hash Function (UHF)** (es. GHASH).
2. **Pseudo Random Function (PRF)** (es. AES).

Una UHF (Keyed Hash Function) $H_k(msg)$ è "universale" se è difficile per un attaccante **che non conosce la chiave** k trovare una collisione $H_k(M_1) = H_k(M_2)$. A differenza degli hash crittografici, se si conosce la chiave k, trovare una collisione **può** essere facile.

La costruzione WC base è:

$$\text{Tag} = \text{UHF}_{k1}(msg) \oplus \text{PRF}_{k2}(\text{IV})$$

Questo produce un **MAC randomizzato**, dove l'IV è usato come input per la PRF per generare un pad "usa e getta". AES-GCM è un caso specifico di questa costruzione.

10 Problema Critico di GCM: Riutilizzo del Nonce (IV)

Il riutilizzo del nonce (IV) in AES-GCM è **catastrofico**.

Ricordiamo la formula del tag:

$$\text{tag} = \text{GHASH}_H(CT) \oplus AES_K(IV, 0)$$

Se un attaccante ottiene due messaggi (CT_1, CT_2) cifrati con lo **stesso IV**:

$$\text{tag}_1 = \text{GHASH}_H(CT_1) \oplus AES_K(IV, 0)$$

$$\text{tag}_2 = \text{GHASH}_H(CT_2) \oplus AES_K(IV, 0)$$

Il pad $AES_K(IV, 0)$ è identico. L'attaccante può calcolare lo XOR dei tag (che conosce):

$$\text{tag}_1 \oplus \text{tag}_2 = (\text{GHASH}_H(CT_1) \oplus AES_K(IV, 0)) \oplus (\text{GHASH}_H(CT_2) \oplus AES_K(IV, 0))$$

$$\text{tag}_1 \oplus \text{tag}_2 = \text{GHASH}_H(CT_1) \oplus \text{GHASH}_H(CT_2)$$

Dato che GHASH è **lineare** (basato su moltiplicazione polinomiale), questa equazione ($tag_1 \oplus tag_2 = GHASH_H(CT_1 \oplus CT_2)$) rivela informazioni sulla chiave di autenticazione H.

Un attaccante può **recuperare la chiave** H e, da quel momento, **falsificare messaggi validi** (creare tag validi per qualsiasi ciphertext). (Almeno la chiave di cifratura K rimane segreta).

10.1 Soluzioni

Esistono modalità robuste al riutilizzo del nonce?. Sì, ad esempio **AES-GCM-SIV** (Synthetic IV), standardizzato in RFC 8452 (Aprile 2019).