OCB: A Bock-Cipher Mode of Operation for Efficient Authenticated Encryption

Phillip Rogaway
UC Davis
This work done at Chiang Mai University
rogaway@cs.ucdavis.edu
http://www.cs.ucdavis.edu/~rogaway

Mihir Bellare
UC San Diego

John Black
University of Nevada

Ted Krovetz
UC Davis

MIT - November 9, 2001
Principal Goals of Symmetric Cryptography

**Privacy**  What the Adversary sees tells her nothing of significance about the underlying message $M$ that the Sender sent.

**Authenticity**  The Receiver is sure that the string he receives was sent (in exactly this form) by the Sender.

**Authenticated Encryption**  Achieves both *privacy* and *authenticity*.
Why Authenticated Encryption?

- **Efficiency**
  By merging privacy and authenticity one can achieve efficiency difficult to achieve if handling them separately.

- **Easier-to-correctly-use abstraction**
  By delivering strong security properties one may minimize encryption-scheme misuse.
Easier to correctly use because stronger security properties

- **Idealized encryption**
- **Authenticated encryption**
  - IND-CPA + auth of ciphertexts
- **IND-CCA = NM-CCA**
- **IND-CPA**
- **OCB**
- **CTR, CBC**
- **ECB**

[Bellare, Rogaway]
[Katz, Yung]
[Bellare, Namprempre]
[Goldwasser, Micali]
[Bellare, Desai, Jokipii, Rogaway]
Right or Wrong?
It depends on what definition $E$ satisfies

\[ A^K \xrightarrow{A \cdot R_A} B^K \]

$E_K(A \cdot B \cdot R_A \cdot R_B \cdot sk)$

$E_K(R_B)$
Generic Composition

Traditional approach to authenticated encryption

Glue together an encryption scheme (E) and a message authentication code (MAC)

Preferred way to do generic composition:
Generic Composition

+ Versatile, clean approach
+ Reduces design work
+ Quick rejection of forged messages if use optimized MAC (eg., UMAC)
+ Inherits the characteristics of the modes one builds from

- Cost $\approx$ (cost to encrypt) + (cost to MAC)
  For CBC Enc + CBC MAC, cost $\approx 2 \times$ (cost to CBC Enc)
- Often done wrong
- Two keys
- Inherits characteristics of the modes one builds from
Trying to do Better

• Numerous attempts to make privacy + authenticity cheaper.
• One approach: stick with generic composition, but find cheaper encryption schemes or MACs.
• Make authenticity an “incidental” adjunct to privacy within a conventional-looking mode:
  • CBC-with-various-checksums (wrong)
  • PCBC in Kerberos (wrong)
  • PCBC of [Gligor, Donescu 99] (wrong)
  • [Jutla 00] First correct solution
• Jutla described two modes, IACBC and IAPM.
• A lovely start, but many improvements possible.
• OCB: inspired by IAPM, but many new characteristics.
Additional Related Work

• [Halevi]—improved on Jutla’s IAPM proof and helped to clarify what was going on in the scheme.

• [Gligor, Donescu]—Proposed IACBC-like scheme, using mod $2^n$ addition.
What is OCB?

- Authenticated-encryption scheme
- Uses any block cipher (eg. AES)
- Computational cost \( \approx \) cost of CBC
- OCB-AES good in SW or HW
- Lots of nice characteristics designed in:
  - Uses \( \lceil |M| / n \rceil + 2 \) block-cipher calls
  - Uses any nonce (needn’t be unpredictable)
  - Works on messages of any length
  - Creates minimum-length ciphertext
  - Uses a single block-cipher key, each block-cipher keyed with it
  - Quick key setup – suitable for single-message sessions
  - Essentially endian-neutral
  - Fully parallelizable
  - No \( n \)-bit additions
- Provably secure: if you break OCB-AES you’ve broken AES
- In IEEE 802.11 draft standard (wireless LANs)
Checksum = $M[1] \oplus M[2] \oplus \cdots \oplus M[m-1] \oplus C[m]0^* \oplus \text{Pad}$

$L = E_K(0)$
Gray-Code Trick

Instead of forming \( L, 2L, 3L, 4L, \ldots \) (in GF(2^n)) we form \( L, 3L, 2L, 6L, \ldots \)

\[
i \quad \gamma(i) \quad ntz(i) \quad 2^{ntz(i)} \quad \gamma(i-1) \oplus 2^{ntz(i)} \quad \text{dec}
\]

<table>
<thead>
<tr>
<th>i</th>
<th>( \gamma(i) )</th>
<th>ntz(i)</th>
<th>( 2^{ntz(i)} )</th>
<th>( \gamma(i-1) \oplus 2^{ntz(i)} )</th>
<th>dec</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0001</td>
<td>0</td>
<td>0001</td>
<td>0001</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0011</td>
<td>1</td>
<td>0010</td>
<td>0011</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>0010</td>
<td>0</td>
<td>0001</td>
<td>0010</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>0110</td>
<td>2</td>
<td>0100</td>
<td>0110</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>0111</td>
<td>0</td>
<td>0001</td>
<td>0111</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>0101</td>
<td>1</td>
<td>0010</td>
<td>0101</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>0100</td>
<td>0</td>
<td>0001</td>
<td>0100</td>
<td>4</td>
</tr>
<tr>
<td>8</td>
<td>1100</td>
<td>3</td>
<td>1000</td>
<td>1100</td>
<td>12</td>
</tr>
</tbody>
</table>

In this way, the \( i^{th} \) point in the sequence is formed by xoring the prior one with \( 2^{ntz(i)} L \). The values \( L(i)=2^{ntz(i)} L \) can be precomputed.

(But Schroeppel points out that one can also do this with \( L, 2L, 3L, \ldots \) by using different \( L(i) \) values.)
Pseudocode of OCB[E, τ]

**algorithm** OCB-Encrypt \(_K\) (Nonce, \(M\))

1. \(L(0) = E_K(0)\)
2. \(L(-1) = \text{lsb}(L(0))? \ (L(0) \gg 1) \oplus \text{Const43} : (L(0) \gg 1)\)
3. **for** \(i = 1, 2, \ldots\) **do**
   - \(L(i) = \text{msb}(L(i-1))? \ (L(i-1) \ll 1) \oplus \text{Const87} : (L(i-1) \ll 1)\)
4. Partition \(M\) into \(M[1] \ldots M[m]\) // each \(n\) bits, except \(M[m]\) may be shorter
5. \(\text{Offset} = E_K(\text{Nonce} \oplus L(0))\)
6. **for** \(i=1\) **to** \(m-1\) **do**
   - \(\text{Offset} = \text{Offset} \oplus L(\text{ntz}(i))\)
   - \(C[i] = E_K(M[i] \oplus \text{Offset}) \oplus \text{Offset}\)
7. \(\text{Offset} = \text{Offset} \oplus L(\text{ntz}(m))\)
8. \(\text{Pad} = E_K(\text{len}(M[m]) \oplus \text{Offset} \oplus L(-1))\)
9. \(C[m] = M[m] \oplus (\text{first} \mid M[m] \mid \text{bits of Pad})\)
10. \(\text{Checksum} = M[1] \oplus \cdots \oplus M[m-1] \oplus C[m]0^* \oplus \text{Pad}\)
11. \(\text{Tag} = \text{first} \tau \text{bits of } E_K(\text{Checksum} \oplus \text{Offset})\)

**return** \(C[1] \ldots C[m] \parallel \text{Tag}\)
Wrong variant #1

Eliminate “post-whitening”
Wrong variant #2

\[ \text{Checksum} = M[1] \oplus M[2] \oplus \ldots \oplus M[m-1] \oplus M[m]0^* \oplus \text{Pad} \]
Wrong variant #3

\[ M[1] + L + E_{K1} \rightarrow C[1] \]
\[ \cdots \]
\[ M[m-1] + (m-1)L + E_K \rightarrow C[m-1] \]
\[ M[m] + mL + E_K \rightarrow \text{Checksum} \]

Nonce

L

2L

(m-1)L

mL

(m+1)L

Pad

chop \tau

Tag
Assurance via Provable Security

• Provable security begins with [Goldwasser, Micali 82]
• Despite the name, one doesn’t really prove security
• Instead, one gives reductions: theorems of the form
  \[
  \text{If } \text{a certain primitive is secure} \\
  \text{then the scheme based on it is secure.}
  \]
  Eg:
  \[
  \text{If } \text{AES is a secure block cipher} \\
  \text{then OCB-AES is a secure authenticated-encryption scheme.}
  \]
  Equivalently:
  \[
  \text{If some adversary A does a good job at breaking OCB-AES} \\
  \text{then some comparably efficient B does a good job to break AES.}
  \]
• Actual theorems quantitative: they measure how much security is “lost” across the reduction.
(Provable security \perp
symmetric/asymmetric)
Privacy
IND$-CPA: Indistinguishability from Random Bits

\[
\text{Adv}_\text{priv}^\text{priv} (A) = \Pr[A^{\text{Real}} = 1] - \Pr[A^{\text{Rand}} = 1]
\]

[Goldwasser, Micali]
[Bellare, Desai, Jokipii, Rogaway]
Authenticity

Authenticity of Ciphertexts

Adversary $A$ **forges** if she outputs $\text{Nonce } C$ s.t.

- $C$ is **valid** (it decrypts to a message, not to invalid)
- there was no earlier query $\text{Nonce } M_i$ that returned $C$

$\text{Adv}^{\text{auth}}(A) = \Pr[A \text{ forges}]$
Block-Cipher Security
PRP and Strong PRP

\[ \text{Adv}_{\text{prp}}(B) = \Pr[B^{E_K} = 1] - \Pr[B^{\pi} = 1] \]
\[ \text{Adv}_{\text{sprp}}(B) = \Pr[B^{E_K E_K^{-1}} = 1] - \Pr[B^{\pi \pi^{-1}} = 1] \]
OCB Theorems

Privacy theorem:

Suppose $\exists$ an adversary $A$ that distinguishes $\text{OCB}[E, \tau]$ in:

- $\text{time} = t$
- $\text{total-num-of-blocks} = \sigma$
- $\text{adv} = \text{Adv}^{\text{priv}}(A)$

Then $\exists$ an adversary $B$ that breaks block cipher $E$ with:

- $\text{time} \approx t$
- $\text{num-of-queries} \approx \sigma$
- $\text{Adv}^{\text{prp}}(B) \approx \text{Adv}^{\text{priv}}(A) - 1.5 \sigma^2 / 2^n$

Authenticity theorem:

Suppose $\exists$ an adversary $A$ that forges $\text{OCB}[E, \tau]$ with:

- $\text{time} = t$
- $\text{total-num-of-blocks} = \sigma$
- $\text{adv} = \text{Adv}^{\text{auth}}(A)$

Then $\exists$ an adversary $B$ that breaks block cipher $E$ with:

- $\text{time} \approx t$
- $\text{num-of-queries} \approx \sigma$
- $\text{Adv}^{\text{sprp}}(B) \approx \text{Adv}^{\text{auth}}(A) - 1.5\sigma^2/2^n - 2^{-\tau}$
Proof Idea

• As usual, focus on the information-theoretic setting OCB[Perm(n),τ]
• **Privacy:** reasonably clear. Every nonce gives a “random” R, and then we pre-whiten with L+R, 2L+R, 3L+R, … , giving
  With high probability, none of these \( X[i] \) values repeat.
• **Authenticity:** much more difficult. Suppose forge using
  nonce \( N, \) ciphertext \( C[1] \cdots C[c] \)
    • **Case 1:** no earlier \((N, \cdot)\)
    • **Case 2:** earlier \((N, C'[1] \cdots C'[m])\) with \( m \neq c \)
    • **Case 3:** earlier \((N, C'[1] \cdots C'[c])\) and
      \[ C[i] \neq C[i] \] for some \( i < c \)
    • **Case 4:** earlier \((N, C'[1] \cdots C'[c])\) and
      \[ C[i] = C[i] \] for all \( i < c \) and \(|C[c]| = |C'[c]|\)
    • **Case 5:** earlier \((N, C'[1] \cdots C'[c])\) and
      \[ C[i] = C[i] \] for all \( i < c \) and \(|C[c]| \neq |C'[c]|\)
Structure Lemma

If $A$ makes $q$ queries of aggregate length $\sigma$ blocks then forges a $c$-block message,

$$\text{Adv}_{\text{OCB}[\text{Perm}(n),\tau]}^{\text{auth}}(A) \leq \max \left\{ \right.$$

$$\sum_{i} \text{Mcoll}\left( m_{i} \right) + \sum_{i<j} \text{MMcoll}\left( m_{i}, m_{j} \right) +$$

$$\sum_{i} \text{CMcoll}\left( c, m_{i} \right) \right\} + \frac{(\sigma + 2q + 5c + 11)^2}{2^{n+1}} + \frac{1}{2^{\tau}}$$
Mcoll, MMcoll, CMcoll (informally)

**Mcoll** (m): Choose a string $N M[1] \ldots M[m] \Sigma$. Choose $L, R \leftarrow \{0,1\}^n$. What’s the chance of a collision when you form all the induced $X[i]$ values (including $0, N+L$)?

**MMcoll** (m, m’): Choose strings $N M[1] \ldots M[m] \Sigma$ and $N' M'[1] \ldots M'[m'] \Sigma'$, choose $L, R, R' \leftarrow \{0,1\}^n$. What’s the chance that one of the $X[i]$ values associated to the first message is the same as an $X'[j]$ value associated to the second?

**CMcoll** (c, m): Choose strings $N M[1] \ldots M[m] C[1] \ldots C[m]$ and $N' C'[1] \ldots C'[c]$. Choose all random values needed to define $X'[c+1]$—the value that $E_K$ determines the tag from. What’s the chance that $X'[c+1]$ collides with another $X[i], X'[j]$?
What the structure lemma does for us

• Eliminates adaptivity as an issue.
• Lets us focus on pairs of messages instead of all $q$ messages.
• Lets us calculate, carry out case analysis.

Proving the structure lemma

• Uses the “game substitution approach” (as in [KR]).
• Six games are used, slowly blending.
Assembly Speed

Data from **Helger Lipmaa**  www.tcs.hut.fi/~helger  helger@tcs.hut.fi

// Best Pentium AES code known.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Speed (cpb)</th>
<th>Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>OCB-AES</td>
<td>16.9</td>
<td>271</td>
</tr>
<tr>
<td>CBC-AES</td>
<td>15.9</td>
<td>255</td>
</tr>
<tr>
<td>ECB-AES</td>
<td>14.9</td>
<td>239</td>
</tr>
<tr>
<td>CBCMAC-AES</td>
<td>15.5</td>
<td>248</td>
</tr>
</tbody>
</table>

6.5 % slower

1 Kbyte messages—pure Pentium 3 assembly—AES128.
Overhead so small that AES with a C-code CBC wrapper is slightly more expensive than AES with an assembly OCB wrapper.

---

C Speed

Data from **Ted Krovetz**. Compiler is MS VC++. Uses rijndael-alg-fst.c ref code.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Speed (cpb)</th>
<th>Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>OCB-AES</td>
<td>28.1</td>
<td>449</td>
</tr>
<tr>
<td>CBCMAC-AES</td>
<td>26.8</td>
<td>428</td>
</tr>
</tbody>
</table>

4.9 % slower
Why I like OCB 😊

• **Ease-of-correct-use.** Reasons: all-in-one approach; any type of nonce; parameterization limited to block cipher and tag length

• **Aggressively optimized:** ≈ optimal in many dimensions: key length, ciphertext length, key setup time, encryption time, decryption time, available parallelism; SW characteristics; HW characteristics; …

• **Simple but non-obvious**

• Ideal setting for **practice-oriented provable security**

For More Information

  Contains FAQ, papers, reference code, …