OCB Mode

**Phillip Rogaway**
Department of Computer Science
UC Davis + Chiang Mai Univ
rogaway@cs.ucdavis.edu
http://www.cs.ucdavis.edu/~rogaway
+66 1 530 7620  +1 530 753 0987

**Mihir Bellare**
UCSD
mihir@cs.ucsd.edu

**John Black**
UNR
jrb@cs.unr.edu

**Ted Krovetz**
Digital Fountain
tdk@acm.org
Looking—contact Ted!

NIST Modes of Operation Workshop 2 – Aug 24, 2001 - Santa Barbara, California
Two Cryptographic Goals

**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**

---

Slide 2
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
What does Encryption Do?

Strong

- Idealized encryption
- Authenticated encryption: IND under CPA + auth of ciphertexts
- IND under CCA = NM under CCA

Weak

- CTR, CBC
- ECB
- No meaningful notion of privacy

Security community’s favored view

Cryptographic community’s favored view: sym encryption is for IND-CPA (and nothing more)

[Bellare, Desai, Jokipii, Rogaway]

[Bellare, Nampremre], [Katz, Yung]
Right or Wrong?

It depends on what definition $E$ satisfies

$$A^K \xrightarrow{A \cdot R_A} B^K$$

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

$$E_K (R_B)$$
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:

Folklore approach. See [Bellare, Namprempre] and [Krawczyk] for analysis.
Generic Composition

+ Versatile, clean architecture
+ Reduces design work
+ Quick rejection of forged messages if use optimized MAC (e.g., 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 misused
- 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 privacy algorithm and cheaper authenticity algorithms
• 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 - Aug 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
What is OCB?

- Authenticated-encryption scheme
- Uses any block cipher (eg. AES)
- Computational cost ≈ cost of CBC
- OCB-AES good in SW or HW
- Lots of nice characteristics designed in:
  - Uses ⌈|M|/n⌉ + 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. Paper to appear at ACM CCS ’01
Checksum = $M[1] \oplus M[2] \oplus \cdots \oplus M[m-1] \oplus C[m]0^* \oplus \text{Pad}$

$Z[i] = Z[i-1] \oplus L(\text{ntz}(i))$

$L(0) = E_K(0)$ and each $L(i)$ obtained from $L(i-1)$ by a shift and conditional xor
Definition of OCB[E, t]

algorithm OCB-Encrypt \( K \) (Nonce, \( M \))

\[
L(0) = E_K(0) \\
L(-1) = \text{lsb}(L(0)) \oplus (L(0) \gg 1) \oplus \text{Const43} : (L(0) \gg 1) \\
\textbf{for } i = 1, 2, \ldots \textbf{ do } L(i) = \text{msb}(L(i-1)) \oplus (L(i-1) \ll 1) \oplus \text{Const87} : (L(i-1) \ll 1) \\
\]

Partition \( M \) into \( M[1] \cdots M[m] \) // each \( n \) bits, except \( M[m] \) may be shorter

\[
\text{Offset} = E_K(\text{Nonce} \oplus L(0)) \\
\textbf{for } i=1 \textbf{ to } m-1 \textbf{ do } \\
\hspace{1em} \text{Offset} = \text{Offset} \oplus L(\text{ntz}(i)) \\
\hspace{1em} C[i] = E_K(M[i] \oplus \text{Offset}) \oplus \text{Offset} \\
\text{Offset} = \text{Offset} \oplus L(\text{ntz}(m)) \\
\text{Pad} = E_K(\text{len}(M[m]) \oplus \text{Offset} \oplus L(-1)) \\
C[m] = M[m] \oplus (\text{first } | M[m] | \text{ bits of Pad}) \\
\text{Checksum} = M[1] \oplus \cdots \oplus M[m-1] \oplus C[m]0^* \oplus \text{Pad} \\
\text{Tag} = \text{first } \tau \text{ bits of } E_K(\text{Checksum} \oplus \text{Offset}) \\
\textbf{return } C[1] \cdots C[m] \| \text{Tag}
\]
## Assembly Speed

Data from **Helger Lipmaa**  
[www.tcs.hut.fi/~helger](http://www.tcs.hut.fi/~helger)  
helger@tcs.hut.fi  
// Best Pentium AES code known. Helger’s code is for sale, btw.

<table>
<thead>
<tr>
<th>Cipher</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>

The above data is for 1 Kbyte messages. Code is pure Pentium 3 assembly.  
The block cipher is 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>Cipher</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>
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 highly non-obvious**
- Ideal setting for **practice-oriented provable security**
What is 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
  \[
  \text{If a certain primitive is secure} \\
  \text{then the scheme based on it is secure}
  \]
Eg:
  \[
  \text{If AES is a secure block cipher} \\
  \text{then OCB-AES is a secure authenticated-encryption scheme}
  \]
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.
The Power of Definitions

• Let’s you carry on an intelligent conversation
• Let’s you investigate the “space” of goals and how they are related
• Often let’s you easily see when protocols are wrong
• Let’s you prove when things are right, to the extent that we know how to do this.

It took about an hour to break the NSA’s “Dual Counter Mode”. What did I have that the NSA authors didn’t? Just an understanding of a good definition for the goal.
Privacy
Indistinguishability from Random Bits

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

**Authenticity of Ciphertexts**

- **A** forges if she outputs forgery attempt $\text{Nonce } C$ s.t.
  - $C$ is valid (it decrypts to a message, not to invalid)
  - there was no $E_K$ 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:

<table>
<thead>
<tr>
<th>Suppose $\exists$ an adversary $A$ that breaks $\text{OCB-E}$ with:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{time} = t$</td>
</tr>
<tr>
<td>$\text{total-num-of-blocks} = \sigma$</td>
</tr>
<tr>
<td>$adv = \text{Adv}^{\text{priv}}(A)$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Then $\exists$ an adversary $B$ that breaks block cipher $E$ with:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{time} \approx t$</td>
</tr>
<tr>
<td>$\text{num-of-queries} \approx \sigma$</td>
</tr>
<tr>
<td>$\text{Adv}^{\text{prp}}(B) \approx \text{Adv}^{\text{priv}}(A) - 1.5 \frac{\sigma^2}{2^n}$</td>
</tr>
</tbody>
</table>

## Authenticity theorem:

<table>
<thead>
<tr>
<th>Suppose $\exists$ an adversary $A$ that breaks $\text{OCB-E}$ with:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{time} = t$</td>
</tr>
<tr>
<td>$\text{total-num-of-blocks} = \sigma$</td>
</tr>
<tr>
<td>$adv = \text{Adv}^{\text{auth}}(A)$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Then $\exists$ an adversary $B$ that breaks block cipher $E$ with:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{time} \approx t$</td>
</tr>
<tr>
<td>$\text{num-of-queries} \approx \sigma$</td>
</tr>
<tr>
<td>$\text{Adv}^{\text{sprp}}(B) \approx \text{Adv}^{\text{priv}}(A) - 1.5 \frac{\sigma^2}{2^n}$</td>
</tr>
</tbody>
</table>
What Provable Security Does, and Doesn’t, Buy You

+ Strong evidence that scheme does what was intended
+ Best assurance cryptographers know how to deliver
+ Quantitative usage guidance

- An absolute guarantee
- Protection from issues not captured by our abstractions
- Protection from usage errors
- Protection from implementation errors
<table>
<thead>
<tr>
<th>Domain</th>
<th></th>
<th>Ciphertext</th>
<th></th>
<th>IV rqmt</th>
<th>Calls / msg</th>
<th>Calls / keysetup</th>
<th>Key length (#E-keys)</th>
<th>blk overhead</th>
<th>E circuit depth</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IAPM</strong> (lazy mod p) [Jutla 00,01]</td>
<td>({0,1}^n)^*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>XECB-XOR</strong> [GD 01]</td>
<td>{0,1}^*</td>
<td>⌈</td>
<td>M</td>
<td>/ n ⌉ + n</td>
<td></td>
<td>ctr</td>
<td></td>
<td></td>
<td>k+2n</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(2)</td>
<td>2 add</td>
<td>3 add</td>
</tr>
<tr>
<td><strong>OCB</strong> [R+ 00,01]</td>
<td>{0,1}^*</td>
<td></td>
<td></td>
<td>nonce</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Parallelizable Authenticated-Encryption Schemes**
For More Information

- OCB web page → www.cs.ucdavis.edu/~rogaway
  Contains FAQ, papers, reference code, licensing info...
- Feel free to call or send email
- Upcoming talks: MIT (Oct 26), ACM CCS (Nov 5-8), Stanford (TBA)
- Or grab me now!

Anything Else ??