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

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**
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

No meaningful notion of privacy

Security community’s favored view

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

OCB

[Bellare, Desai, Jokipii, Rogaway]

CTR, CBC$

[ Bellare, Nampremre], [Katz, Yung]

ECB

[557x700]Slide 4
Right or Wrong?

It depends on what definition $E$ satisfies
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 $\approx$ 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. Paper to appear at ACM CCS ’01
Checksum = $M[1] \oplus M[2] \oplus \ldots \oplus M[m-1] \oplus C[m]0^* \oplus \text{Pad}$

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

$L(0) = E_K(0)$ 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) = lsb(L(0)) \(\oplus\) (L(0) \(\gg\) 1) \(\oplus\) Const43 \(\oplus\) (L(0) \(\gg\) 1)

for \(i = 1, 2, \ldots\) do
L(i) = msb(L(i-1)) \(\oplus\) (L(i-1) \(\ll\) 1) \(\oplus\) Const87 \(\oplus\) (L(i-1) \(\ll\) 1)

Partition M into M[1] \(\ldots\) M[m] // each n bits, except M[m] may be shorter
Offset = E\(_K\) (Nonce \(\oplus\) L(0))

for i=1 to m-1 do
  Offset = Offset \(\oplus\) L(ntz(i))
  C[i] = E\(_K\) (M[i] \(\oplus\) Offset) \(\oplus\) Offset

Offset = Offset \(\oplus\) L(ntz(m))
Pad = E\(_K\) (len(M[m]) \(\oplus\) Offset \(\oplus\) L(-1))
C[m] = M[m] \(\oplus\) (first | M[m] | bits of Pad)
Checksum = M[1] \(\oplus\) \(\ldots\) \(\oplus\) M[m-1] \(\oplus\) C[m]0* \(\oplus\) Pad
Tag = first \(\tau\) bits of E\(_K\) (Checksum \(\oplus\) Offset)

**return** C[1] \(\ldots\) C[m] || Tag
**Assembly Speed**

Data from **Helger Lipmaa**  
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>Cycles</th>
<th>Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>OCB-AES</td>
<td>271</td>
<td>16.9 cpb</td>
</tr>
<tr>
<td>CBC-AES</td>
<td>255</td>
<td>15.9 cpb</td>
</tr>
<tr>
<td>ECB-AES</td>
<td>239</td>
<td>14.9 cpb</td>
</tr>
<tr>
<td>CBCMAC-AES</td>
<td>248</td>
<td>15.5 cpb</td>
</tr>
</tbody>
</table>

6.5% slower

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>Cycles</th>
<th>Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>OCB-AES</td>
<td>449</td>
<td>28.1 cpb</td>
</tr>
<tr>
<td>CBCMAC-AES</td>
<td>428</td>
<td>26.8 cpb</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 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

$E_K(\text{Nonce}_i, M_i)$

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

$\text{Adv}^{\text{auth}}(A) = \Pr[A \text{ forges}]$

[Bellare, Rogaway]
[Katz, Yung]
this paper
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 ∃ an adversary $A$ that breaks $OCB-E$ with:
- $time = t$
- $total-num-of-blocks = \sigma$
- $adv = Adv^{priv}(A)$

Then ∃ an adversary $B$ that breaks block cipher $E$ with:
- $time \approx t$
- $num-of-queries \approx \sigma$
- $Adv^{prp}(B) \approx Adv^{priv}(A) - 1.5 \sigma^2 / 2^n$

Authenticity theorem:

Suppose ∃ an adversary $A$ that breaks $OCB-E$ with:
- $time = t$
- $total-num-of-blocks = \sigma$
- $adv = Adv^{auth}(A)$

Then ∃ an adversary $B$ that breaks block cipher $E$ with:
- $time \approx t$
- $num-of-queries \approx \sigma$
- $Adv^{sprp}(B) \approx Adv^{priv}(A) - 1.5 \sigma^2 / 2^n$
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></th>
<th>Domain</th>
<th>Ciphertext</th>
<th>IV reqmt</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></td>
<td>${0,1}^+$</td>
<td>$</td>
<td>M</td>
<td>+ \tau$</td>
<td>nonce</td>
<td>$</td>
<td>M</td>
<td>/n + 2$</td>
</tr>
<tr>
<td>(lazy mod p)</td>
<td>[Jutla 00,01]</td>
<td></td>
<td>(Jutla’s presentation gave rand version)</td>
<td></td>
<td></td>
<td></td>
<td>2 add</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 addp</td>
<td></td>
</tr>
<tr>
<td><strong>XECB-XOR</strong></td>
<td>${0,1}*$</td>
<td>$\lceil</td>
<td>M</td>
<td>/ n \rceil + n$</td>
<td>ctr</td>
<td>$\lceil</td>
<td>M</td>
<td>/ n \rceil + 1$</td>
</tr>
<tr>
<td>[GD 01]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(1)</td>
<td>3 add</td>
<td></td>
</tr>
<tr>
<td><strong>OCB</strong></td>
<td>${0,1}*$</td>
<td>$</td>
<td>M</td>
<td>+ \tau$</td>
<td>nonce</td>
<td>$\lceil</td>
<td>M</td>
<td>/ n \rceil + 2$</td>
</tr>
<tr>
<td>[R+ 00,01]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(1)</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 ??