The Evolution of Authenticated Encryption

Phillip Rogaway
University of California, Davis, USA

Includes joint work with
Mihir Bellare, John Black, Ted Krovetz, and Tom Shrimpton

DIAC — Directions in Authenticated Ciphers
05 July 2012
Stockholm, Sweden

Today

Historically ordered

1. Introduction

- The recognition of AE as a useful “thing”
- Modes that don’t work

2. Definitions and constructions

- Defining **AE**
- Generic composition
- RPC, XCBC$, IAPM, OCB
- Defining nonce-based **AEAD**
- CCM
- GCM
- OCB, again
- Defining **MRAE**
- SIV

3. Discussion

- Taxonomy
- Patents
- Suggestions
- Sample research questions

Authenticated Encryption (AE)

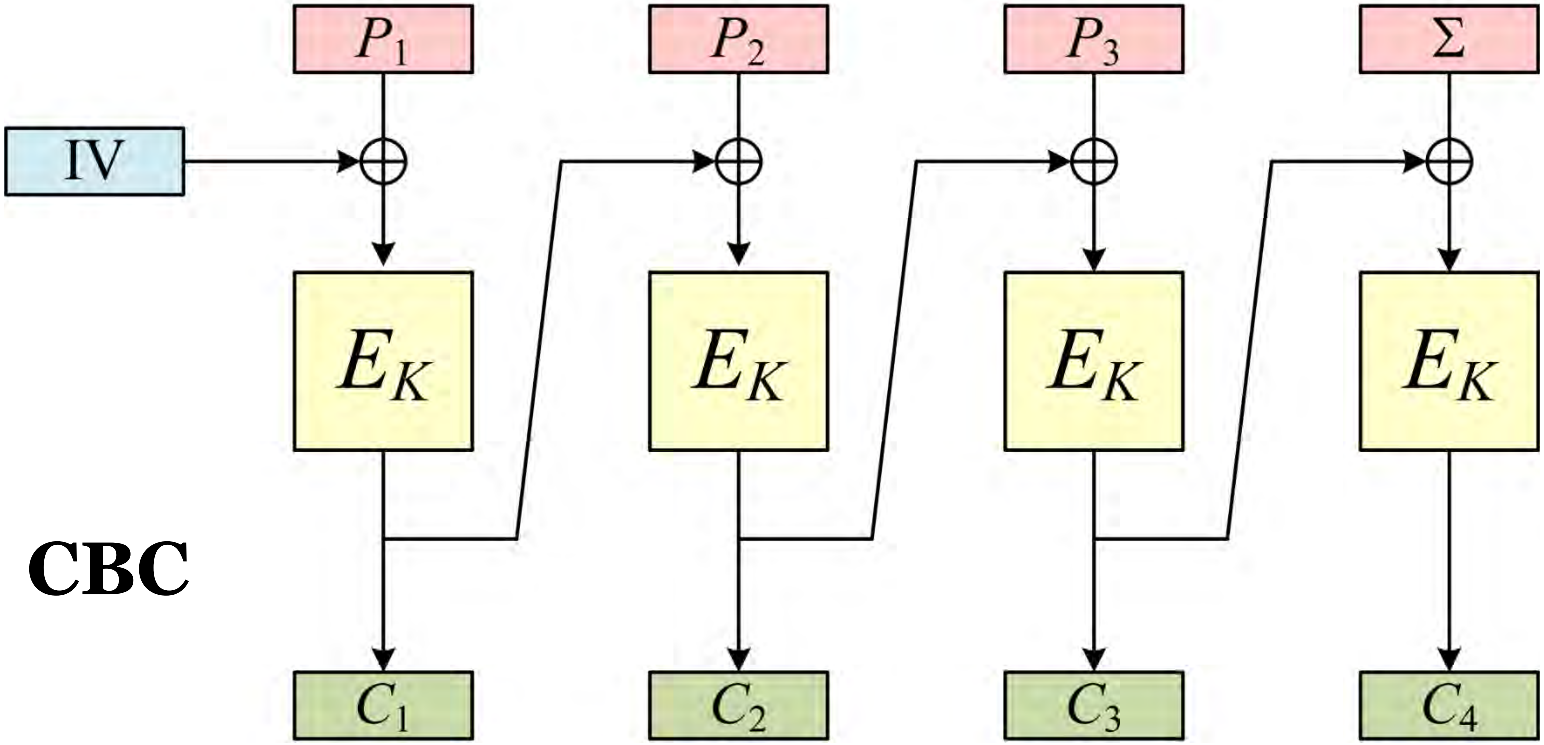
Promises **two benefits**

1. An easier-to-correctly-use abstraction boundary
2. More efficient realizations

Begins with two **realizations** regarding symmetric encryption

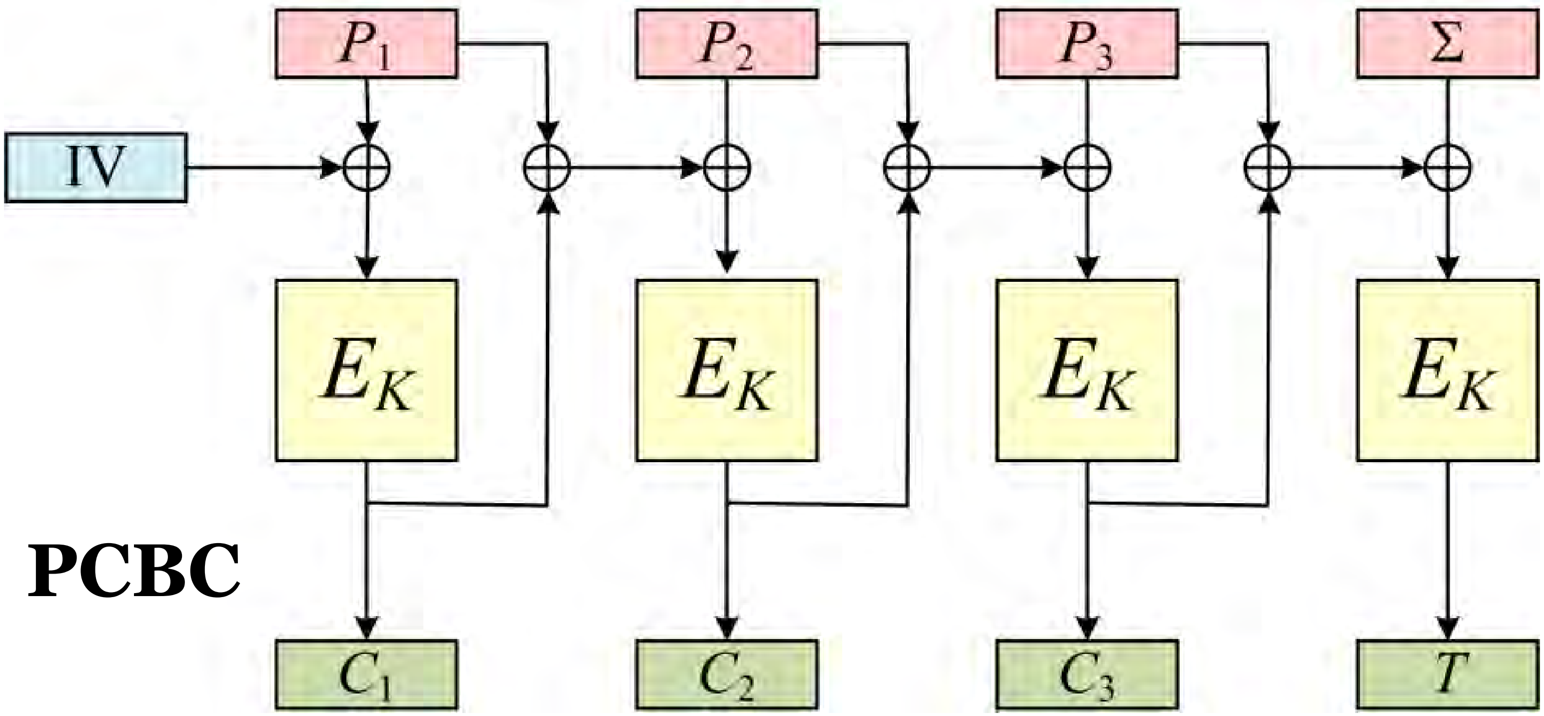
1. "Integrity" /"authenticity" is routinely needed
2. "Standard" privacy mechanisms don't provide it

Check / insert redundancy



CBC

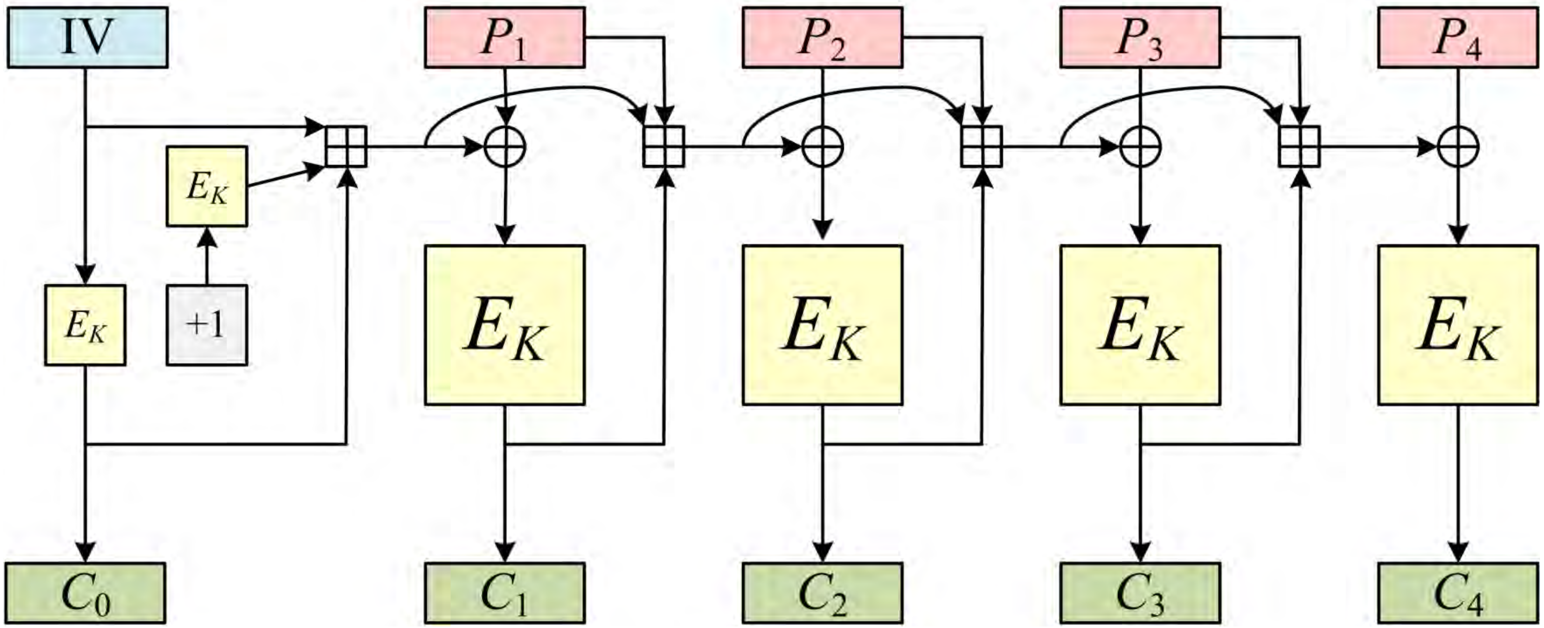
Add more arrows



See: Yu, Hartman, Raeburn 2004
“The Perils of Unauthenticated
Encryption: Kerberos Version 4”

Still more arrows/operations

iaPCBC
[Gligor, Donescu 1999]



Promptly broken by **Jutla** (1999) and by
Ferguson, Whiting, Kelsey, Wagner (1999)

~2000

Emerging understanding that:

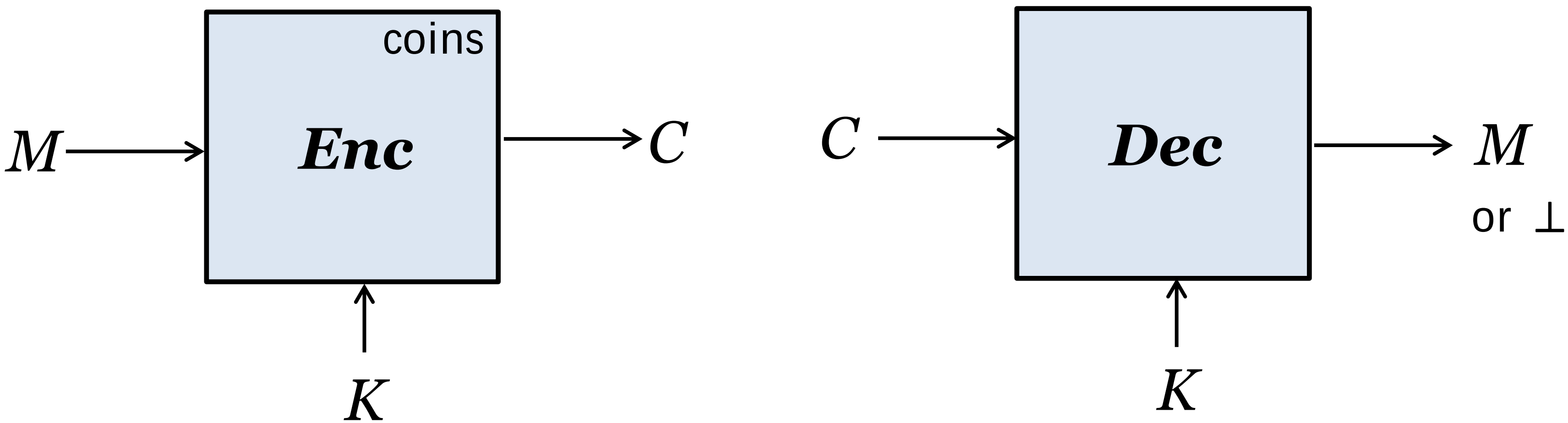
- **Beyond IND-CPA privacy** was often desirable
- Didn't come with **standard** encryption methods
- Simple ways to try to get it **cheaply** don't work

Similar realizations in the public-key world …

- [Bleichenbacher 1998] – "A chosen ciphertext attack against protocols based on the RSA encryption standard PKCS #1"
- Reaction was that IND-CPA security was not enough
  - **CCA1** security (Naor-Yung 1990)
  - **CCA2** security (Rackoff-Simon 1991)
  - **Non-malleability** (Dolev-Dwork-Naor 1991)

AE Defined

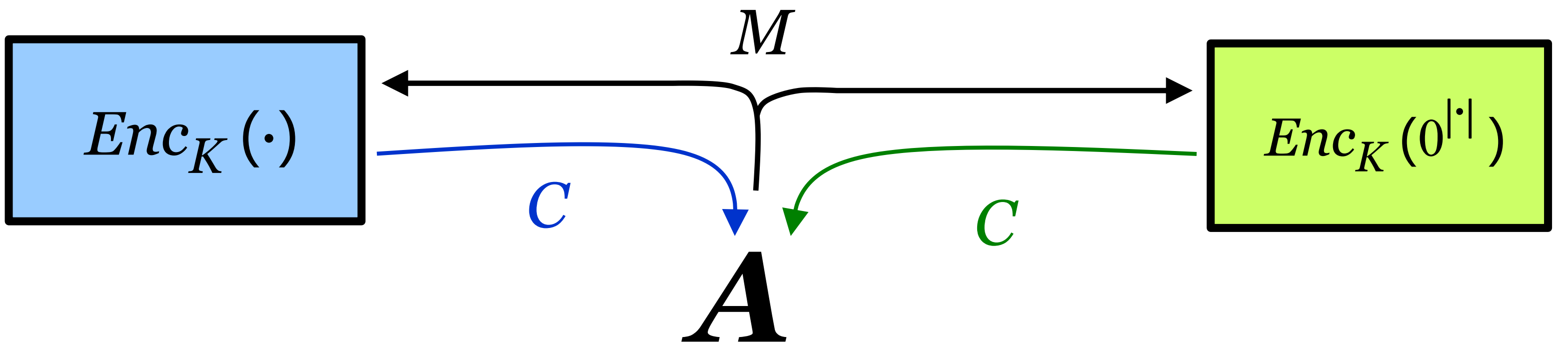
[**Bellare, Rogaway 2000**] – "Encode-then-encipher encryption: how to exploit nonces or redundancy in plaintexts for efficient cryptography"
[**Katz, Yung 2000**] – "Unforgeable encryption and chosen ciphertext secure modes of operation"



- **Conventional privacy** [BDJR97]: Indistinguishability / semantic security.
- **Authenticity**: The only ciphertexts C that will decrypt to something **valid** are those previously obtained by an $\boldsymbol{Enc}(\cdot)$ call.

AE Defined

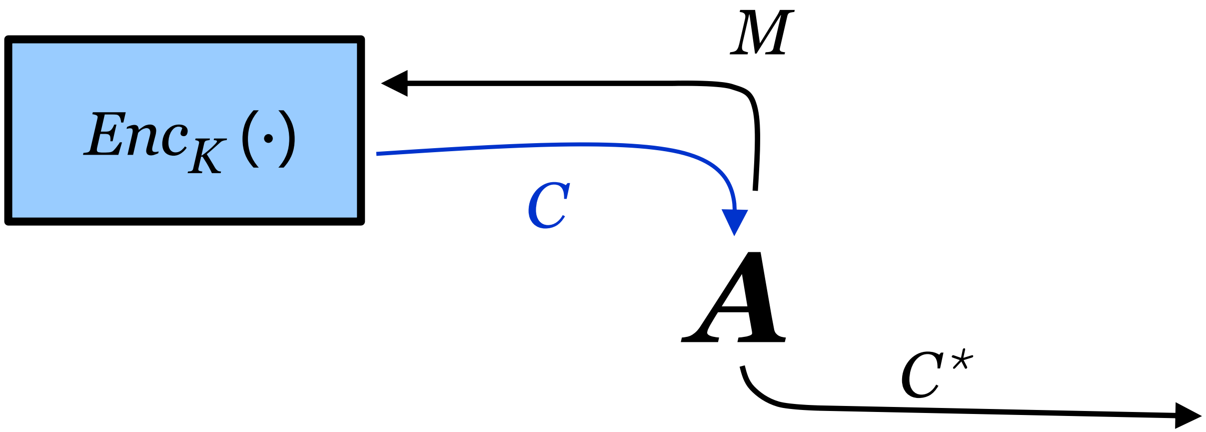
[BDJR97]



$$\mathbf{Adv}_{\Pi}^{\mathrm{priv}}(\boldsymbol{A}) = \Pr[\boldsymbol{A}^{\mathrm{Enc}_K(\cdot)} \rightarrow 1] - \Pr[\boldsymbol{A}^{\mathrm{Enc}_K(0^{|\cdot|})} \rightarrow 1]$$

AE Defined

[BR00,KY00;BN00]



$$\mathbf{Adv}^{\mathrm{priv}}_{\Pi}(\boldsymbol{A}) = \Pr[\boldsymbol{A}^{\mathrm{Enc}_K(\cdot)} \to 1] - \Pr[\boldsymbol{A}^{\mathrm{Enc}_K(0^{|\cdot|})} \to 1]$$

$$\mathbf{Adv}^{\mathrm{auth}}_{\Pi}(\boldsymbol{A}) = \Pr[\boldsymbol{A}^{\mathrm{Enc}_K(\cdot)} \to C^*\text{: no query returned } C^* \text{ and } \mathrm{Dec}_K(C^*) \neq \perp]$$

[BN00, KY0]

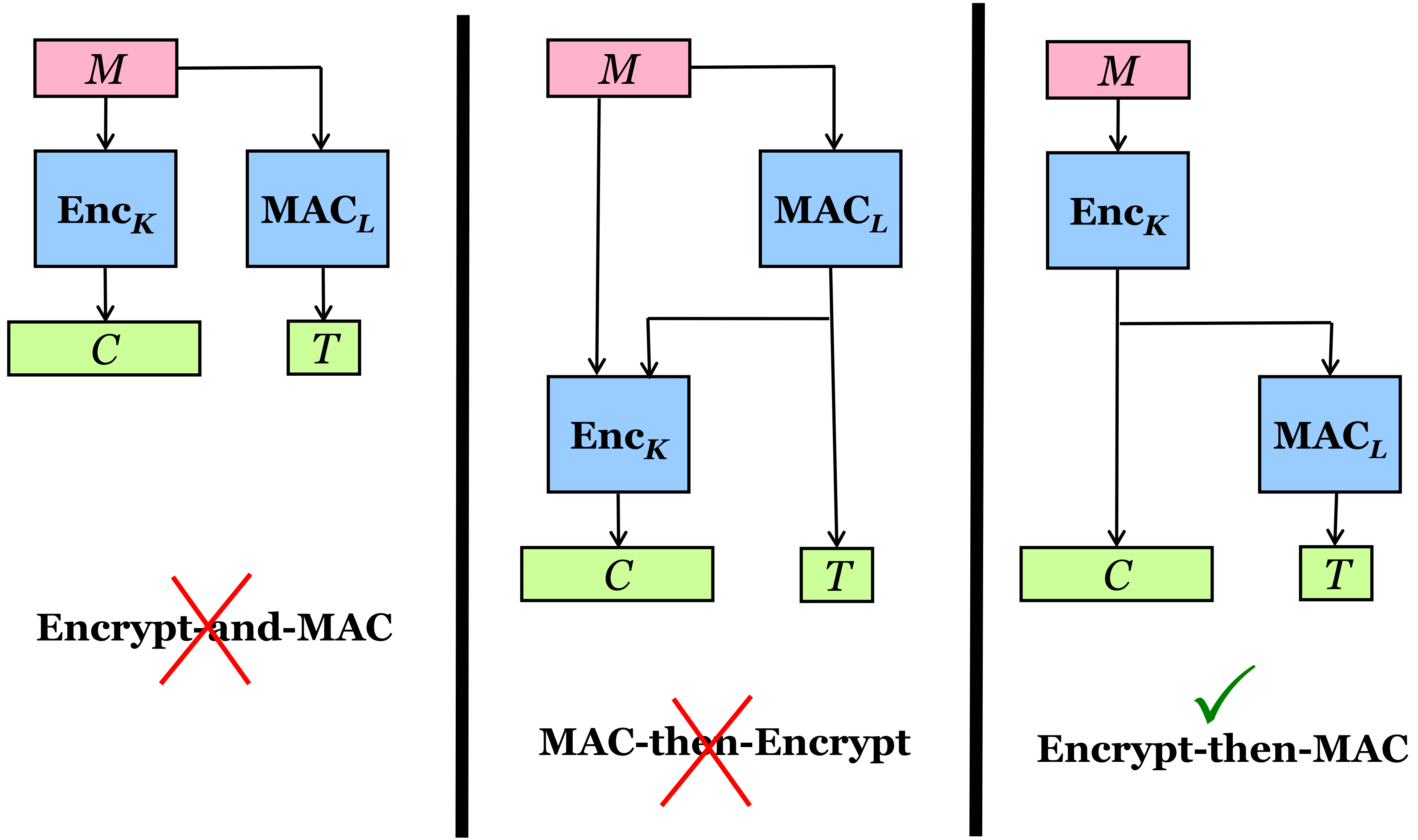
The Strength of AE

- Implies **IND-CCA2** security
- Implies **NM-CCA2** security

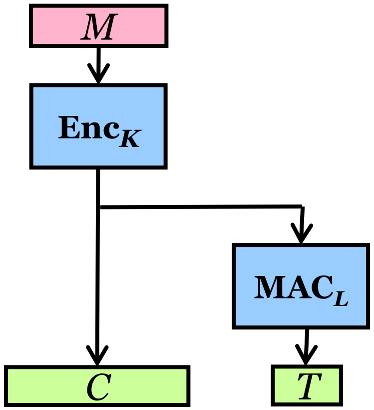
Generic Composition

of an IND-CPA encryption scheme and a PRF

[BN 2000]



The Cost of Generic Composition



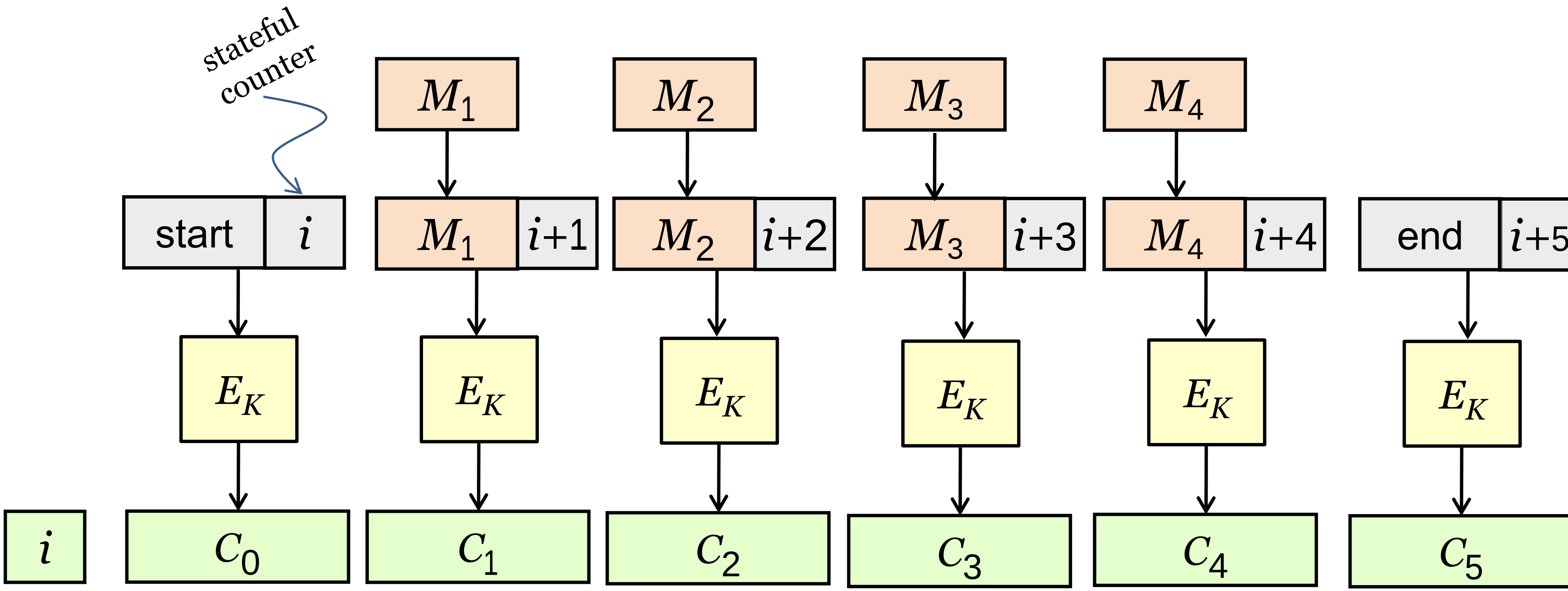
Cost($\mathbf{AE}$) = Cost($\mathbf{Enc}$) + Cost($\mathbf{MAC}$)

Example cases:
Enc = CTR, CBC
MAC = CMAC, HMAC, PMAC, UMAC

➔ Generic composition can be pretty cheap – if you use a cheap MAC

RPC Mode

[KYoo]



XCBC$ Mode

[Gligor Donescu 2001]

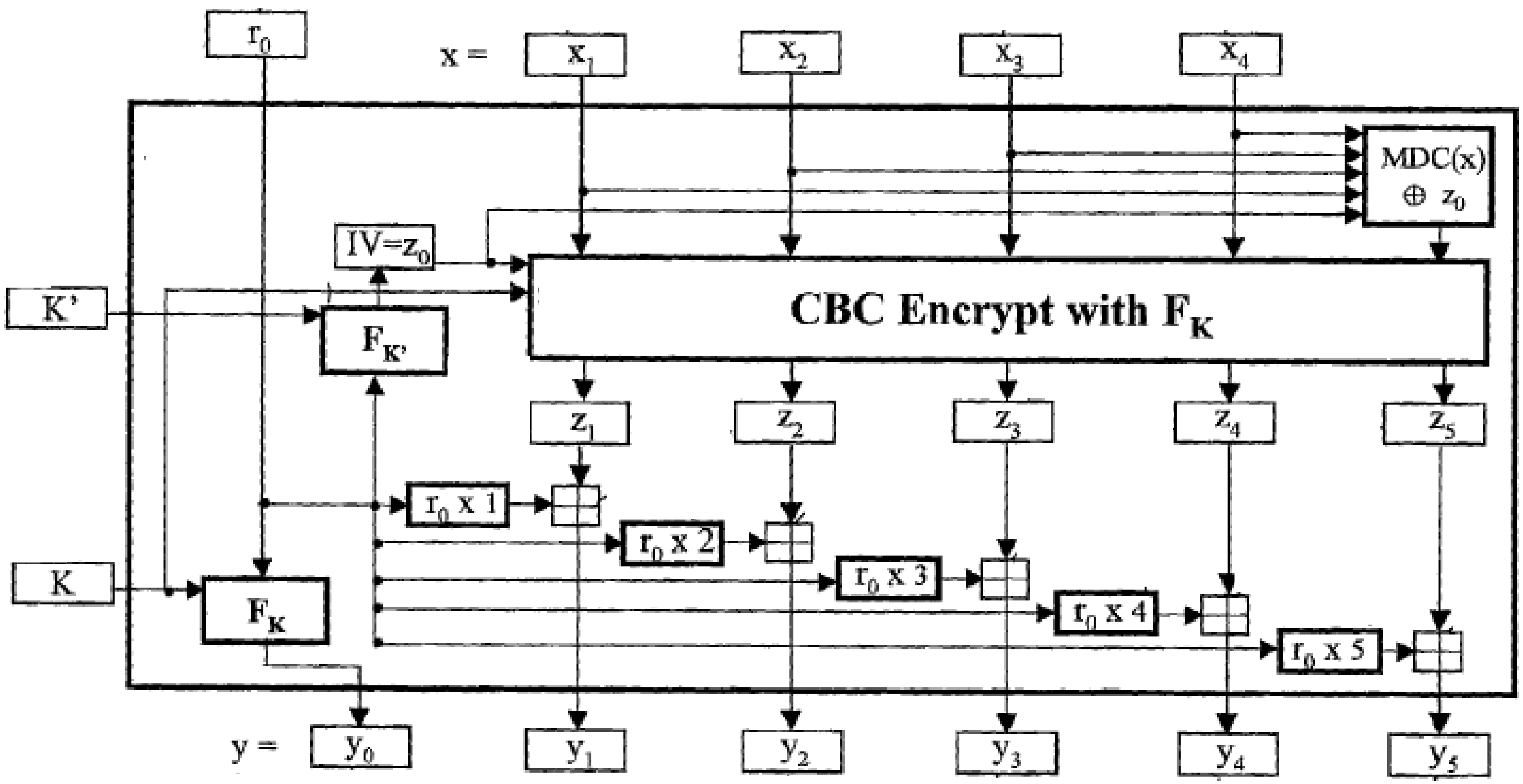


Illustration from
Gligor-Donescu
US Patent 6973182 (2001)

IAPM Mode

[Jutla 2001]

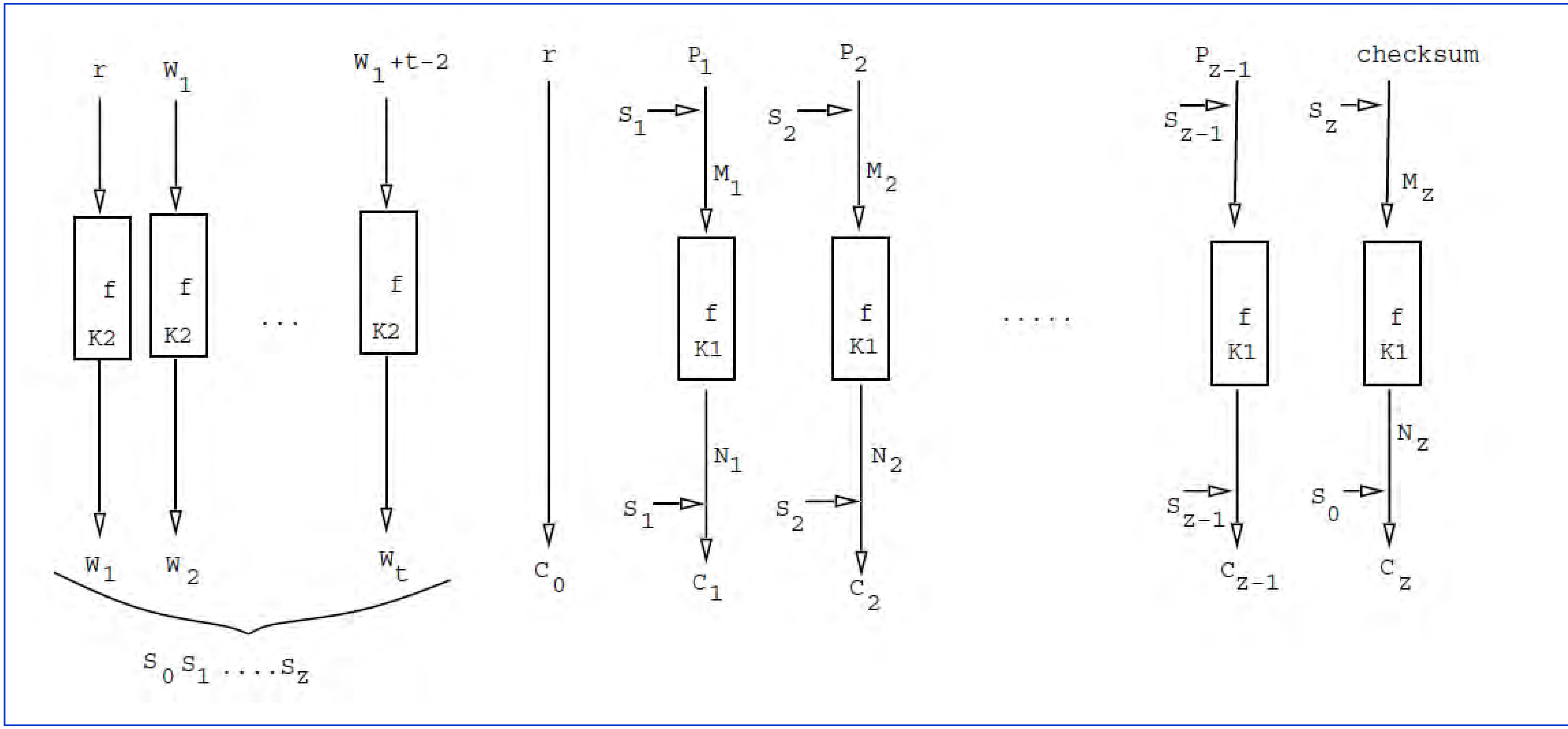
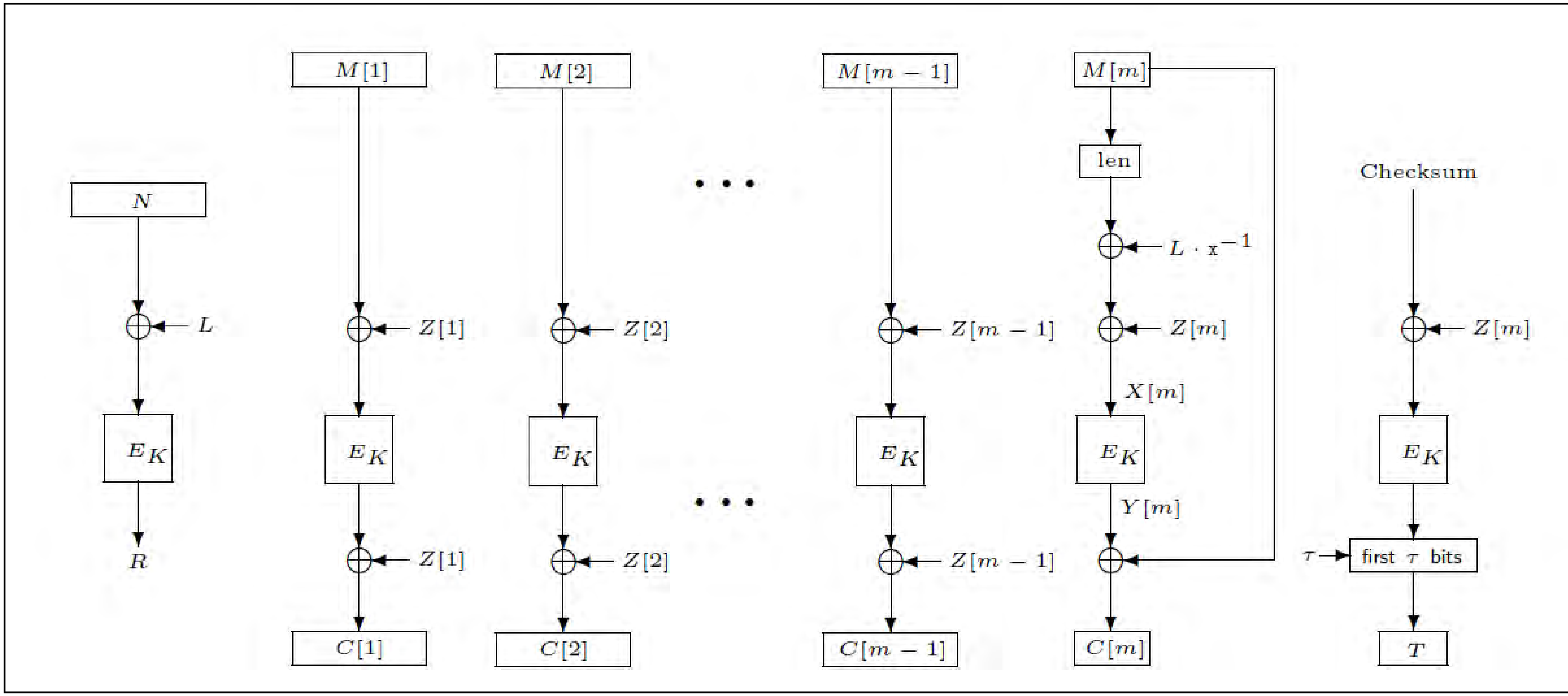


Illustration from
[Jutla 2001]

OCB Mode (later "OCB1")

[R, Bellare, Black, Krovetz 2001]

Like IAPM but highly optimized.
Motivated by NIST's modes call.



$Z[i] = R \oplus \gamma_i \cdot L$
Checksum = $M[1] \oplus \cdots \oplus M[m\text{-}1] \oplus C[m]0^* \oplus Y[m]$

- Arbitrary-length messages
- Efficient offset calculations
- m + 2 blockcipher calls, $m = \lceil |M| / n \rceil$
- Single blockcipher key
- Cheap key setup (one blockcipher call)

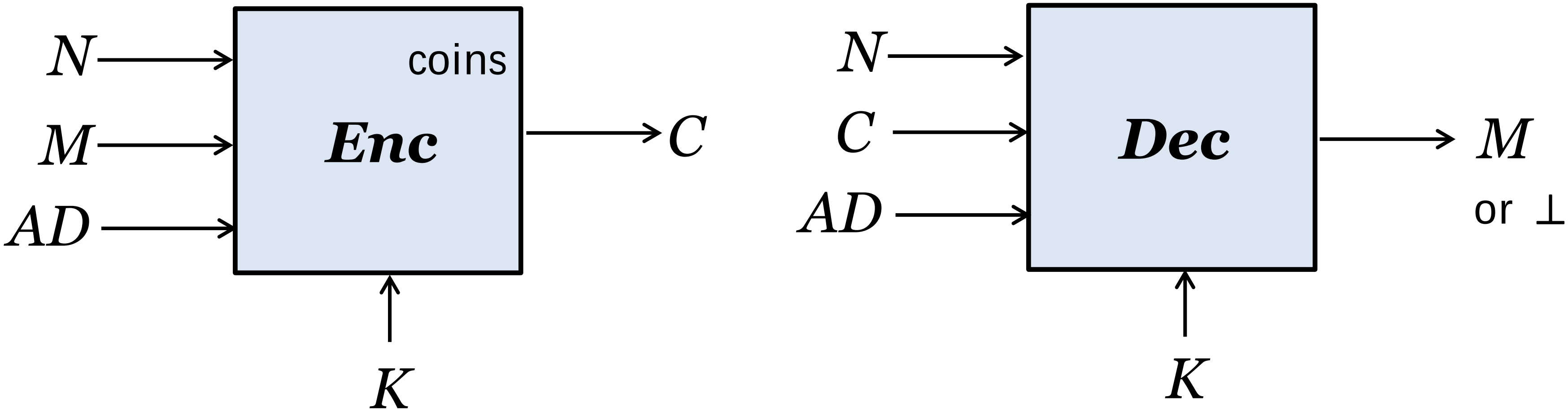
Two important players: NIST and IEEE 802.11i

- **WiFi** standard ratified in 1999
  Uses **WEP** security

- **Fatal attacks** soon emerge:
  - [Fluhrer, Mantin, Shamir 2001]
    Weaknesses in the key scheduling algorithm of RC4
  - [Stubblefield, Ioannidis, Rubin 2001]
    Using the Fluhrer, Mantin, Shamir attack to break WEP
  - [Borisov, Goldberg, Wagner 2001]
    Intercepting mobile communications: the insecurity of 802.11
  - [Cam-Winget , Housley, Wagner, Walker 2003]
    Security flaws in 802.11 data links protocols

- **WEP → TKIP → WPA → WPA2**
  - Draft solutions based on OCB
  - Politics and patent-avoidance:
    [Whiting, Housley, Ferguson 2002] develop **CCM** (=CCMP)
  - CCM standardized for 802.11, then NIST

Before describing CCM …
Back to the definitional story



- Random values routinely aren't
- Many application have an available nonce
- Weaker user requirement; less misuse

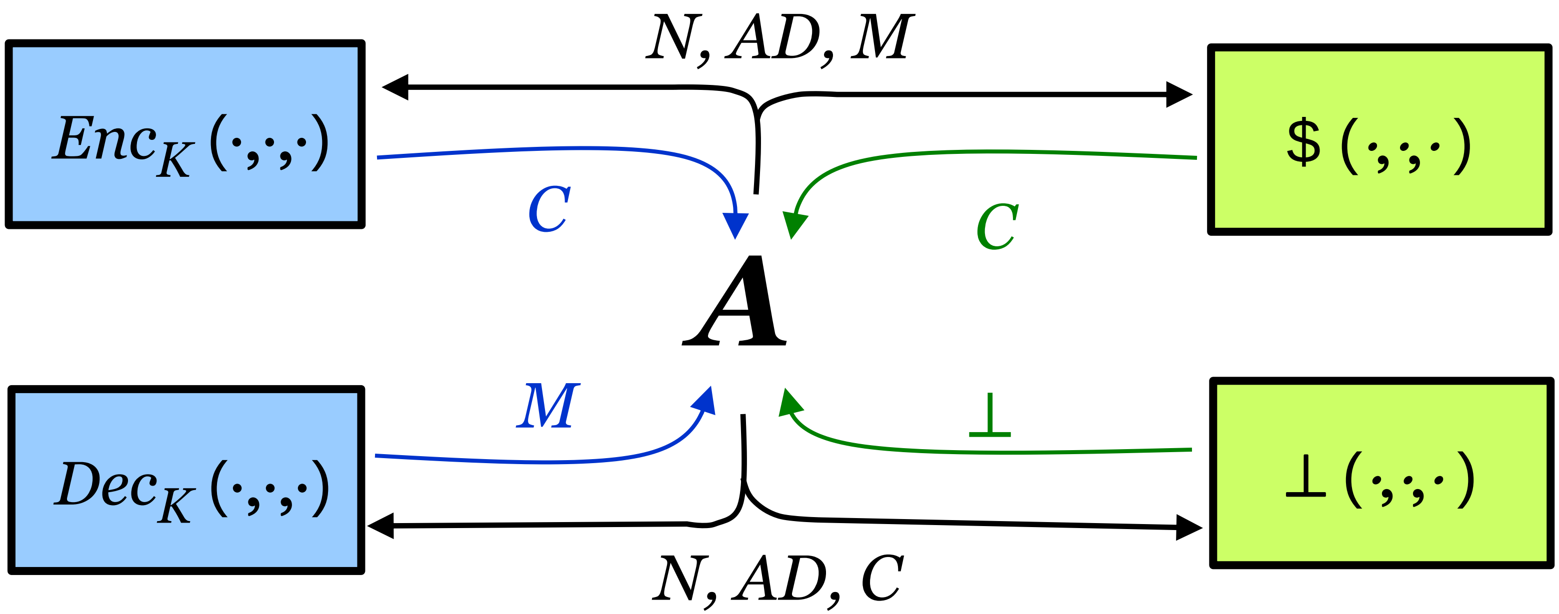
1) Move the coins "out" and make a "nonce" sufficient **[RBBK01]**

2) Add in "associated data" **[R02]**

- Requirement from Cam-Winget, Kaliski, Walker
- AD is authenticated but not encrypted
- Failure to provide same AD on decryption results in ⊥

AEAD

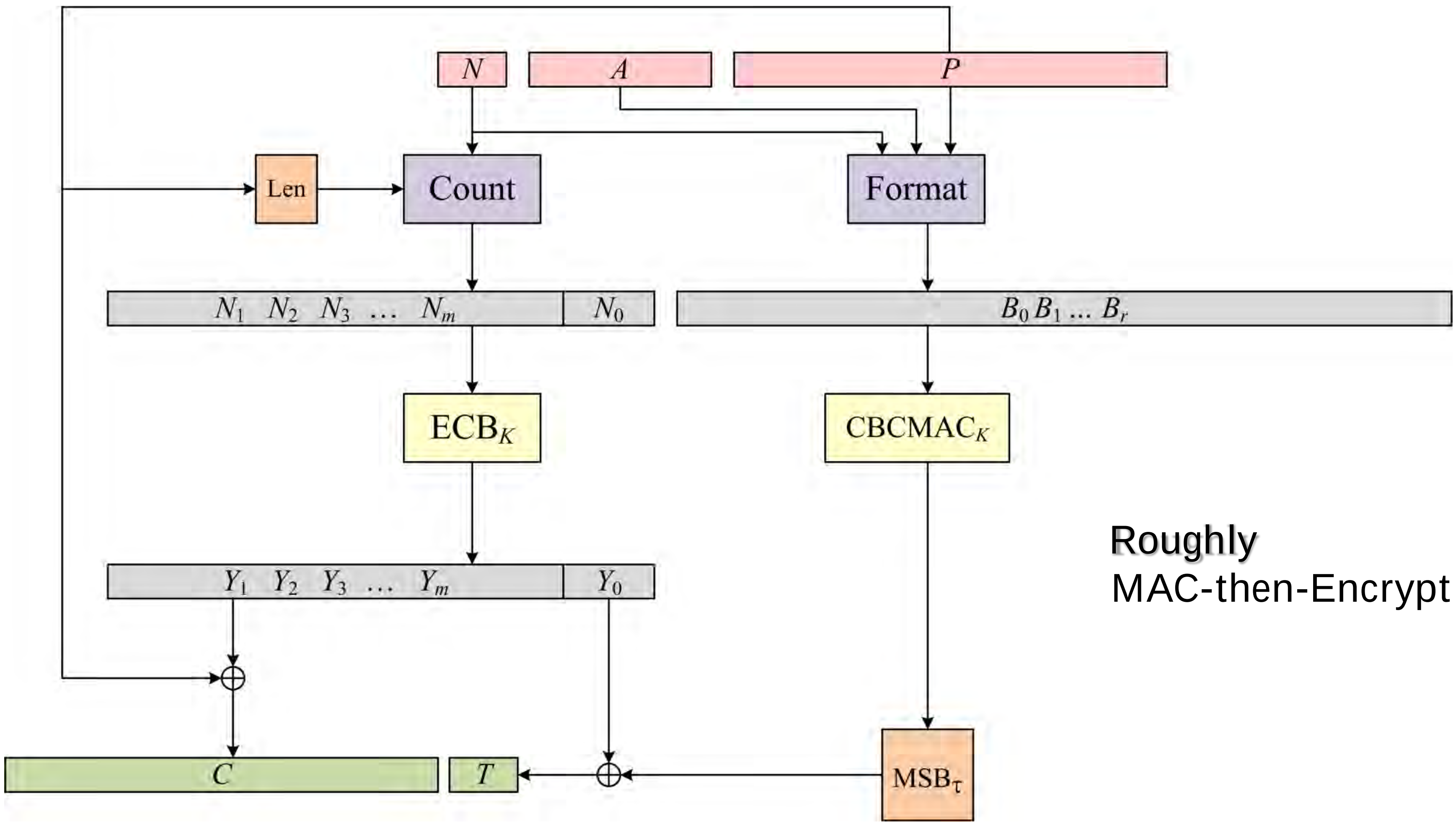
Also: (1) Ask for indistinguishability from random bits **[RBBK00]**
(2) All-in-one definition **[R, Shrimpton 2006]**



$$\mathbf{Adv}_{\Pi}^{\text{aead}}(\boldsymbol{A}) = \Pr[\boldsymbol{A}^{Enc_K\ Dec_K} \rightarrow 1] \;-\; \Pr[\boldsymbol{A}^{\$\ \perp} \rightarrow 1]$$

A may not: repeat an N-value in an enc query; or ask a dec query (N, AD, C) after C is returned by an $(N, AD, \cdot)$ enc query

CCM Mode



Roughly
MAC-then-Encrypt

Functions FORMAT and COUNT

$$\textsc{Count}_q(N,m) = N_1 \parallel N_2 \parallel \cdots \parallel N_m \text{ where}$$

$$N_i = 0^5 \parallel [\,q-1\,]_3 \parallel N \parallel [\,i\,]_{8q}$$

$$\textsc{Format}_{q,t}(N,A,P) =$$

$$0 \parallel \textbf{if } A=\varepsilon \textbf{ then } 0 \textbf{ else } 1 \textbf{ endif} \parallel [\,t/2-1\,]_3 \parallel [\,q-1\,]_3 \parallel$$

$$N \parallel [\,|P|_8\,]_{8q} \parallel$$

$$\textbf{if } A=\varepsilon \textbf{ then } \varepsilon \textbf{ elseif}$$

$$|A|_8 < 2^{16}-2^8 \textbf{ then } [\,|A|_8\,]_{16}$$

$$\textbf{elseif } |A|_8 < 2^{32} \textbf{ then } \mathtt{0xFFFE} \parallel [\,|A|_8\,]_{32} \textbf{ else } \mathtt{0xFFFF} \parallel [\,|A|_8\,]_{64} \textbf{ endif} \parallel$$

$$A \parallel$$

$$\textbf{if } A=\varepsilon \textbf{ then } \varepsilon \textbf{ elseif } |A|_8 < 2^{16}-2^8 \textbf{ then } (\mathtt{0x00})^{(14-|A|_8) \bmod 16}$$

$$\textbf{elseif } |A|_8 < 2^{32} \textbf{ then } (\mathtt{0x00})^{(10-|A|_8) \bmod 16} \textbf{ else } (\mathtt{0x00})^{(6-|A|_8) \bmod 16} \textbf{ endif} \parallel$$

$$P \parallel$$

$$(\mathtt{0x00})^{(-|M|_8) \bmod 16}$$

See: **[R 2011]**, "Evaluation of Some Blockcipher Modes of Operation" (Ch. 11); following **[R, Wagner 2003]**, "A Critique of CCM"

CCM Mode

[Whiting, Housley, Ferguson 2002]
NIST SP 800-38C:2004
RFC 3610, 4309, 5084, 5116

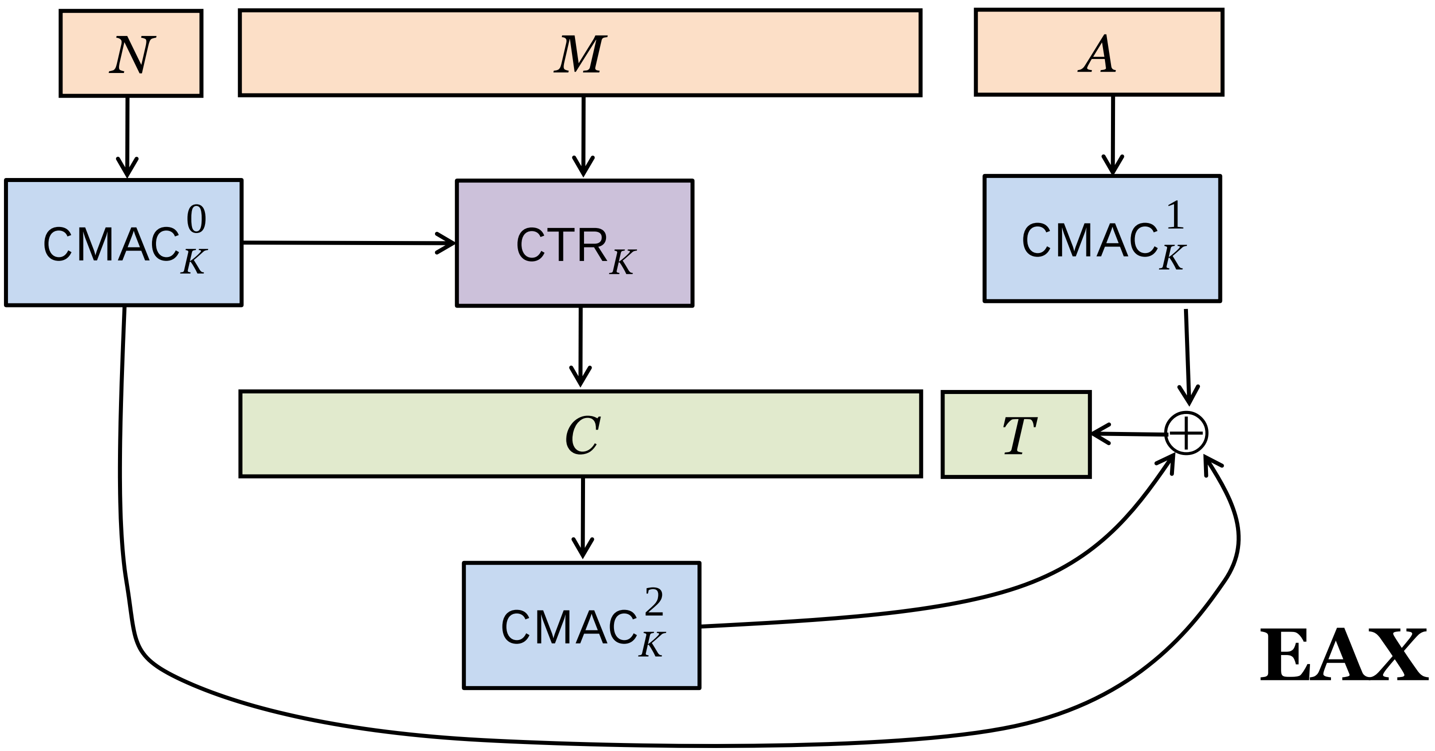
- Provably secure, with OK bounds, if AE if E is a good PRP **[Jonsson 2002]**
- Widely used, standardized (eg, in 802.11)
- Simple to implement
- Only forward direction of blockcipher used

- About $2m+2$ blockcipher calls
- Half non-parallelizable
- Word alignment disrupted
- Can't preprocess static AD
- Not "online" — need to know m in advance
- Complex → Bit twiddling formatting → Absent abstraction boundary
- User must specify

  $q \in \{2,3,4,5,6,7,8\}$ – byte length of byte length of longest message which determines nonce length(!) of $\tau = 15 - q$

The issues with CCM aren’t hard to fix

- Generic composition of CTR and CMAC is a good alternative
- EAX is a CCM-like mode intended to fix CCM’s problems

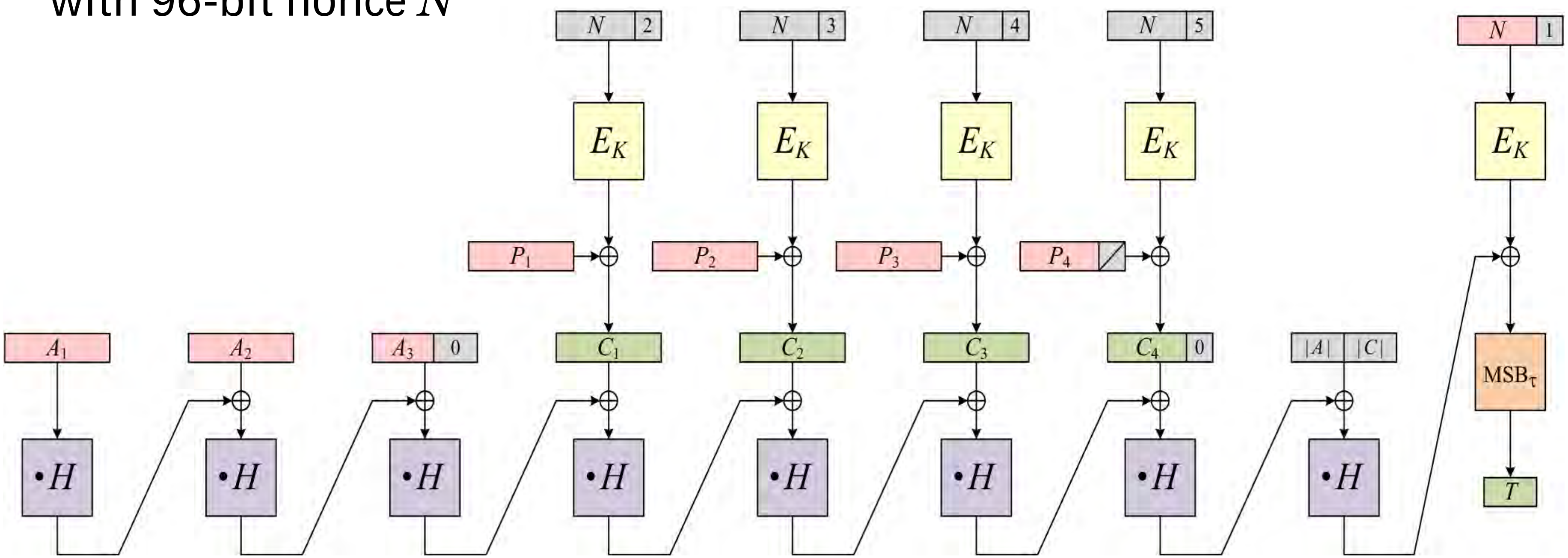


See: **[R 2011]**, “Evaluation of Some Blockcipher Modes of Operation”, Ch. 12

[McGrew, Viega 2004]
(Follows **CWC**
[Kohno, Viega, Whiting 2004])
NIST SP 800-38D:2007
RFC 4106, 5084, 5116, 5288, 5647
ISO 19772:2009

GCM Mode

with 96-bit nonce N



GCM Mode

[McGrew, Viega 2004]
Follows CWC
[Kohno, Viega, Whiting 2004]
NIST SP 800-38D
IPsec, TLS, MACsec, P1619.1, TLS
ISO 19772:2009

- Provably secure, with OK bounds for long tags
- Parallelizable, online
- About m+1 blockcipher calls, all of them parallelizable
- Very efficient in HW
- Reasonably efficient in SW with AES-NI, PCMULDQ, preprocessing & tables
- Static AD can be preprocessed
- Only forward direction of blockcipher used

  - First forgery after $2^{\tau/2}$ queries
  - After, additional forgeries come quickly

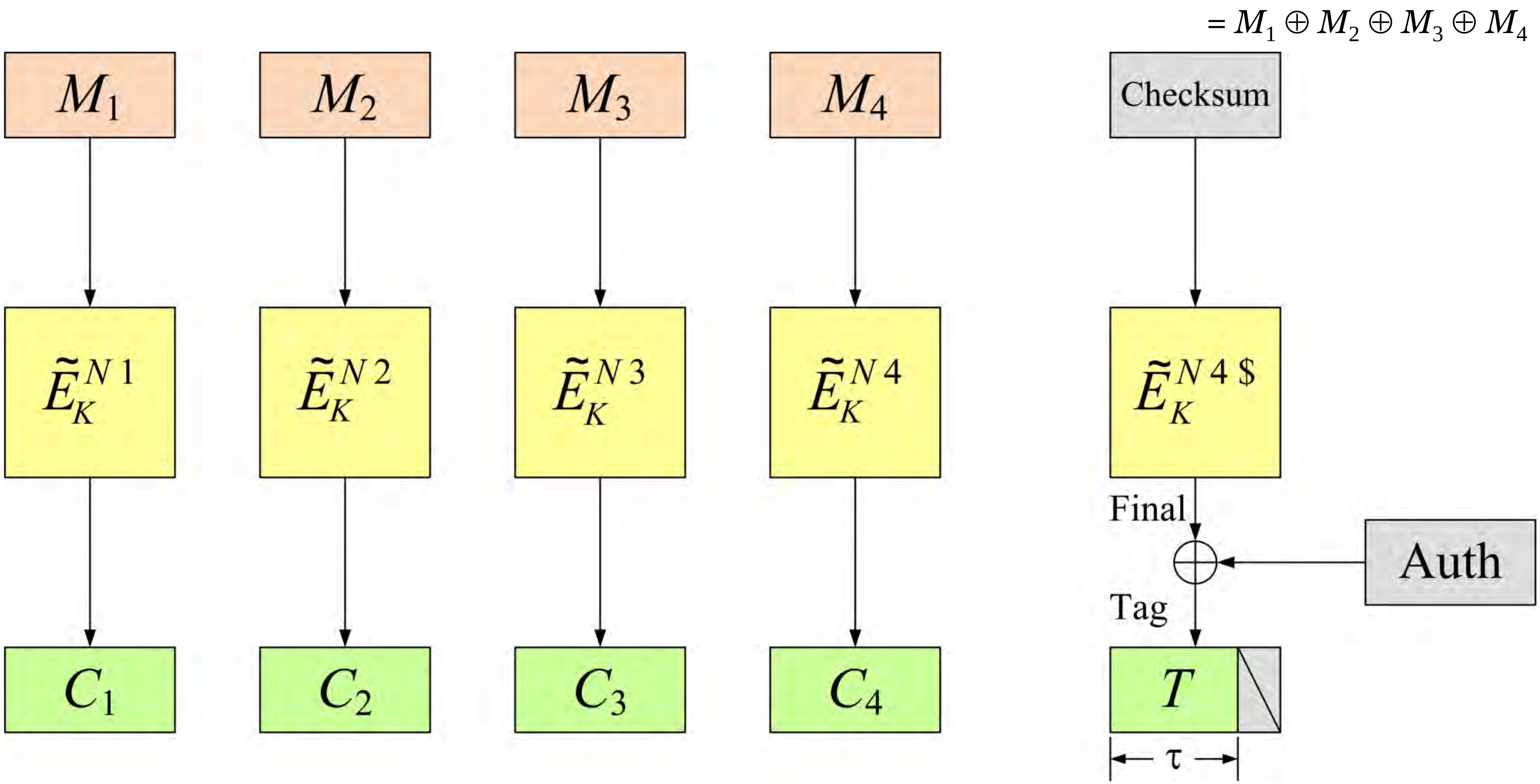
- Poor bound if truncate tag too much **[Ferguson, 2005]** (don't truncate <96 bits)
- Not **that** efficient in SW, even with PCMULDQ support
- Timing attacks an issue for table-based realizations (slow setup, too)
- Maximum of 2^{36}-32 bytes
- "Reflected-bit" convention for representing field points unfortunate
- $|N| \neq 96$ case not handled well
- Published proof is buggy [Iwata, 2012]

OCB3

[RBBK01, R04, KR10]

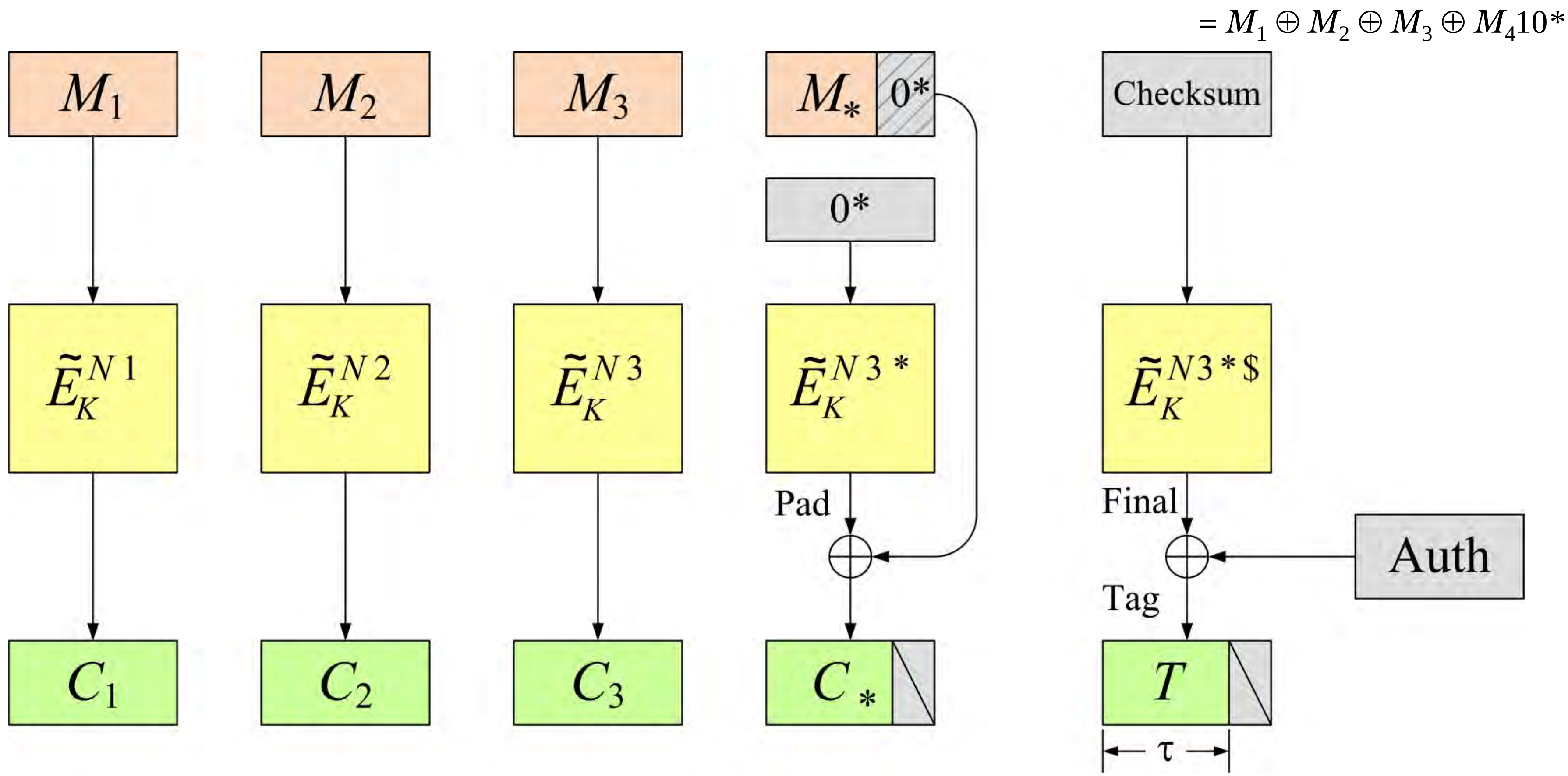
OCB Mode

in terms of a tweakable blockcipher [LRW02]



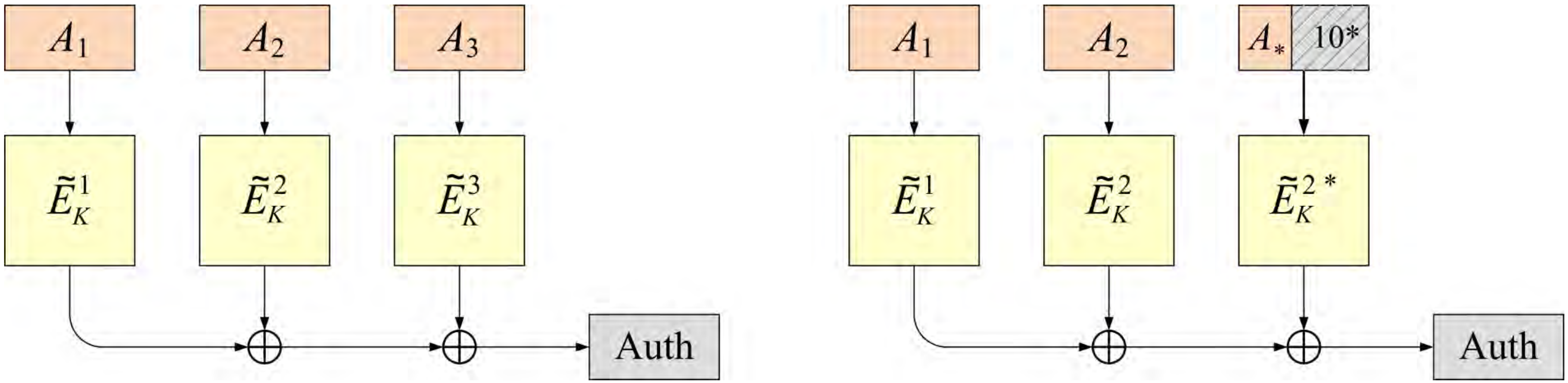
OCB Mode

in terms of a tweakable blockcipher [LRW02]



OCB Mode

in terms of a tweakable blockcipher [LRW02]



Making the Tweakable Blockcipher

$$\widetilde{E}_K^{N\,i}(X) = E_K(X \oplus \Delta) \oplus \Delta \quad \text{with} \quad \Delta = \text{Initial} + \lambda_i L$$

$$\widetilde{E}_K^{N\,i\,*}(X) = E_K(X \oplus \Delta) \quad \text{with} \quad \Delta = \text{Initial} + \lambda_i^* L$$

$$\widetilde{E}_K^{N\,i\,\$}(X) = E_K(X \oplus \Delta) \quad \text{with} \quad \Delta = \text{Initial} + \lambda_i^\$ L$$

$$\widetilde{E}_K^{N\,i\,*\,\$}(X) = E_K(X \oplus \Delta) \quad \text{with} \quad \Delta = \text{Initial} + \lambda_i^{*\$} L$$

$$\widetilde{E}_K^{i}(X) = E_K(X \oplus \Delta) \quad \text{with} \quad \Delta = \lambda_i L$$

$$\widetilde{E}_K^{i\,*}(X) = E_K(X \oplus \Delta) \quad \text{with} \quad \Delta = \lambda_i^* L$$

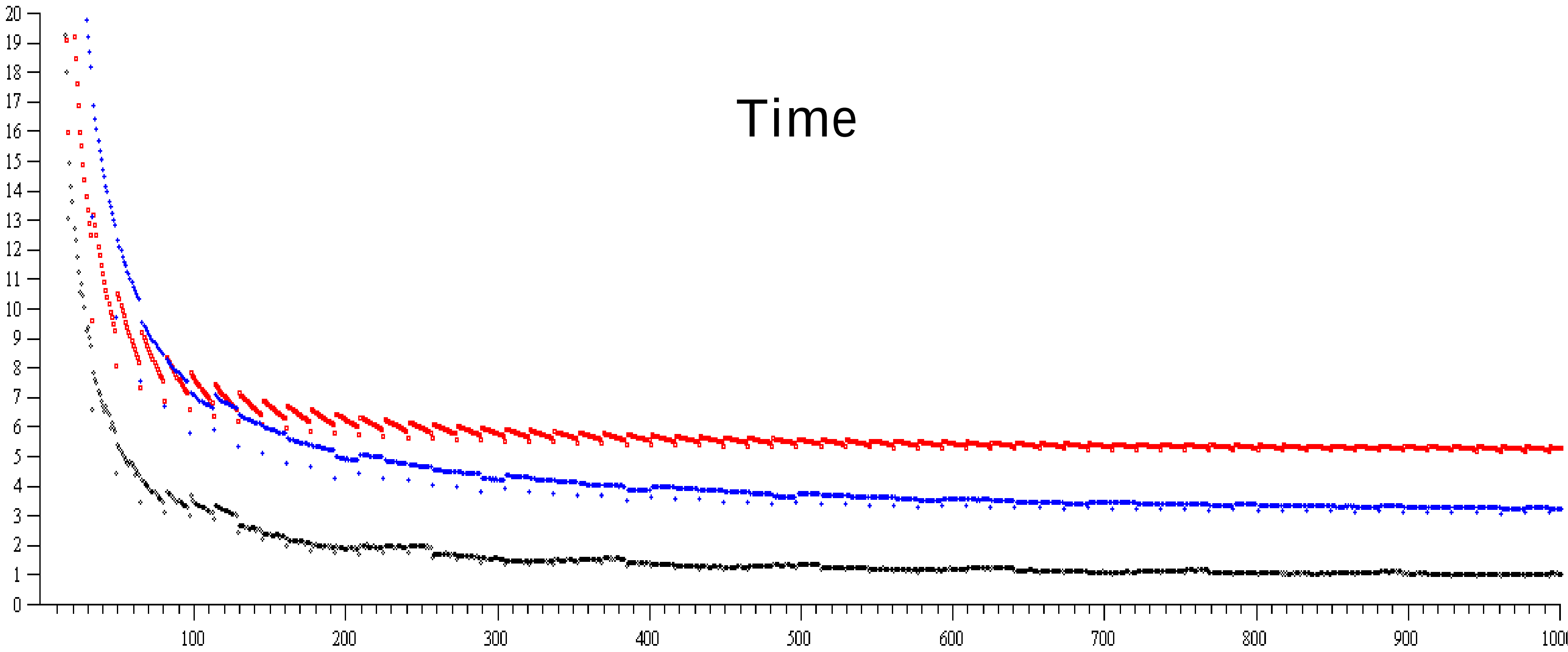
$$\begin{aligned}
\text{Nonce} &= 0^{127-|N|}\, 1\, N \\
\text{Top} &= \text{Nonce} \,\&\, 1^{122}\, 0^6 \\
\text{Bottom} &= \text{Nonce} \,\&\, 1^{122}\, 1^6 \\
\text{Ktop} &= E_K(\text{Top}) \\
\text{Stretch} &= \text{Ktop} \,\|\, (\text{Ktop} \oplus (\text{Ktop} \ll 8)) \\
\text{Initial} &= (\text{Stretch} \ll \text{Bottom})\,[1..128]
\end{aligned}$$

$$\begin{aligned}
L &= E_K(0^{128}) \\
\lambda_i &= 4\, a(i) \\
\lambda_i^* &= 4\, a(i)+1 \\
\lambda_i^\$ &= 4\, a(i)+2 \\
\lambda_i^{*\$} &= 4\, a(i)+3
\end{aligned}$$

$$\begin{aligned}
a(0) &= 0 \\
a(i) &= a(i-1) \oplus 2^{\mathbf{ntz}\,(i)}
\end{aligned}$$

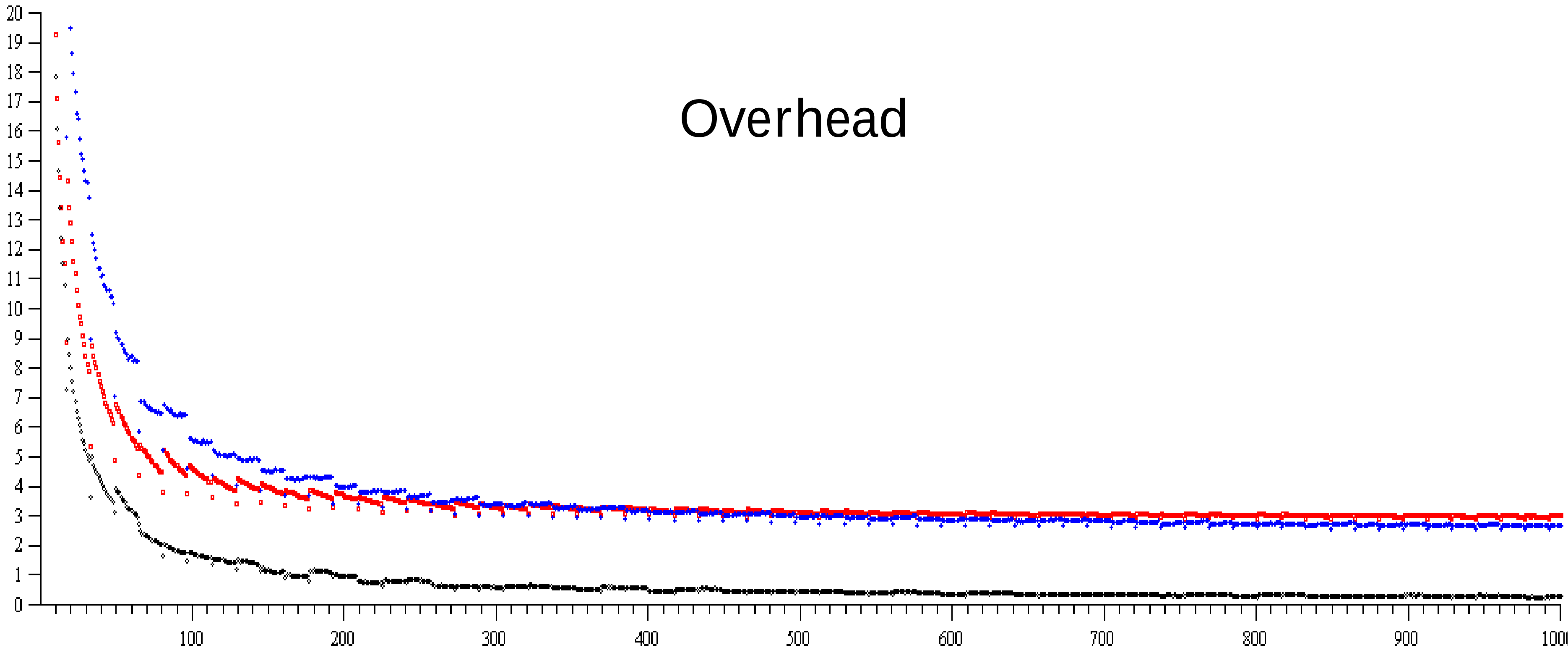
Software Performance
Intel Core x86 i7 – “**Sandy Bridge**”
64-bit OS, using AES/GCM NIs

Mode	4KB cpb
CCM	5.14
GCM	2.95
OCB	0.87



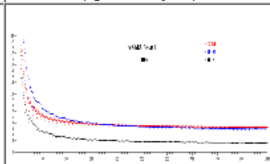
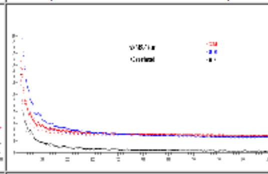
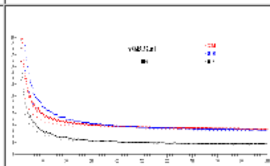
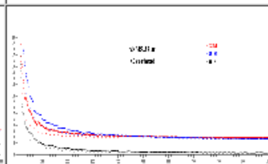
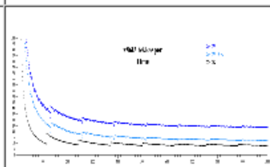
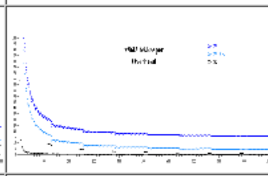
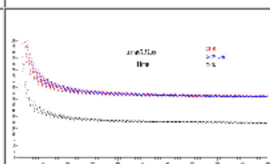
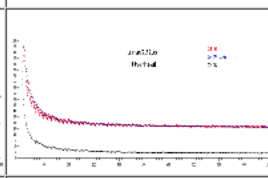
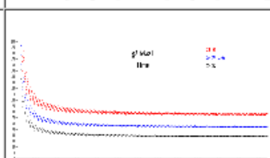
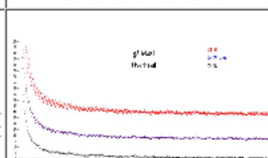
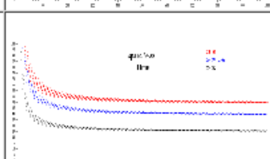
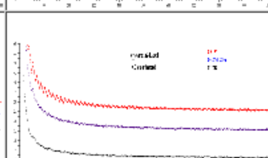
Software Performance
Intel Core x86 i5-650 – "**Clarkdale**"
64-bit OS, using AES/GCM NIs

Mode	4K cpb
CCM	2.09
GCM	2.46
OCB	0.21



Authenticated-Encryption Software Performance: Comparison of CCM, GCM, and OCB

- Click on a **Time** or **Overhead** plot to see a larger version of it.
- Click on a **Mode** (CCM, GCM, OCB, etc) to retrieve the **raw data**.
- Here OCB means OCB3. A companion webpage compares the performance of OCB variants.
- Further notes can be found on the bottom of this page.

Environment (details)	Time (cpb *vs.* bytes)	Overhead (subtract time for CTR)	Mode (clickable)	Over 4096	Time 4096	IPI (cpb)	Size (bytes)	Init (cycles)
Intel x86 i5-650 "Clarkdale" 64-bit NI			CCM GCM OCB CTR	2.90 2.46 0.21	4.17 3.73 1.48 1.27	4.57 4.53 1.87 1.37	512 656 624 244	265 337 295 115
Intel x86 i5-650 "Clarkdale" 32-bit NI			CCM GCM OCB CTR	2.79 2.49 0.20	4.18 3.88 1.59 1.39	4.70 4.79 2.04 1.52	512 656 624 244	274 365 318 130
Intel x86 i5-650 "Clarkdale" 64-bit Käsper-Schwabe			GCM GCM-8K OCB CTR	14.7 3.19 0.31	22.4 10.9 8.05 7.74	26.7 15.2 9.24 8.98	1456 9648 3216 1424	3780 2560 3430 1180
ARM Cortex-A8 32-bit OpenSSL			CCM GCM-256 OCB CTR	25.9 26.7 3.49	51.3 50.8 28.9 25.4	53.7 53.9 30.9 25.9	512 656 784 244	1390 3440 2050 236
PowerPC 970 64-bit OpenSSL			CCM GCM-256 OCB CTR	38.2 16.0 0.0	75.7 53.5 37.5 37.5	77.8 56.2 39.6 37.8	512 656 784 244	1510 1030 2300 309
UltraSPARC III 64-bit OpenSSL			CCM GCM-256 OCB CTR	25.3 15.2 0.9	49.4 39.3 25.0 24.1	51.7 41.5 26.5 24.4	512 656 784 244	1280 904 1770 213

See the OCB homepage for more platforms and data, or for reference code.

Limitations of OCB

1. Blockcipher used in the forward and **backward** direction
2. No CTR-like **locality** in blocks being enciphered
3. Security “only” to **birthday bound**
4. Not resistant to **non-reuse**

Misuse-Resistant AE

[R, Shrimpton 2006]

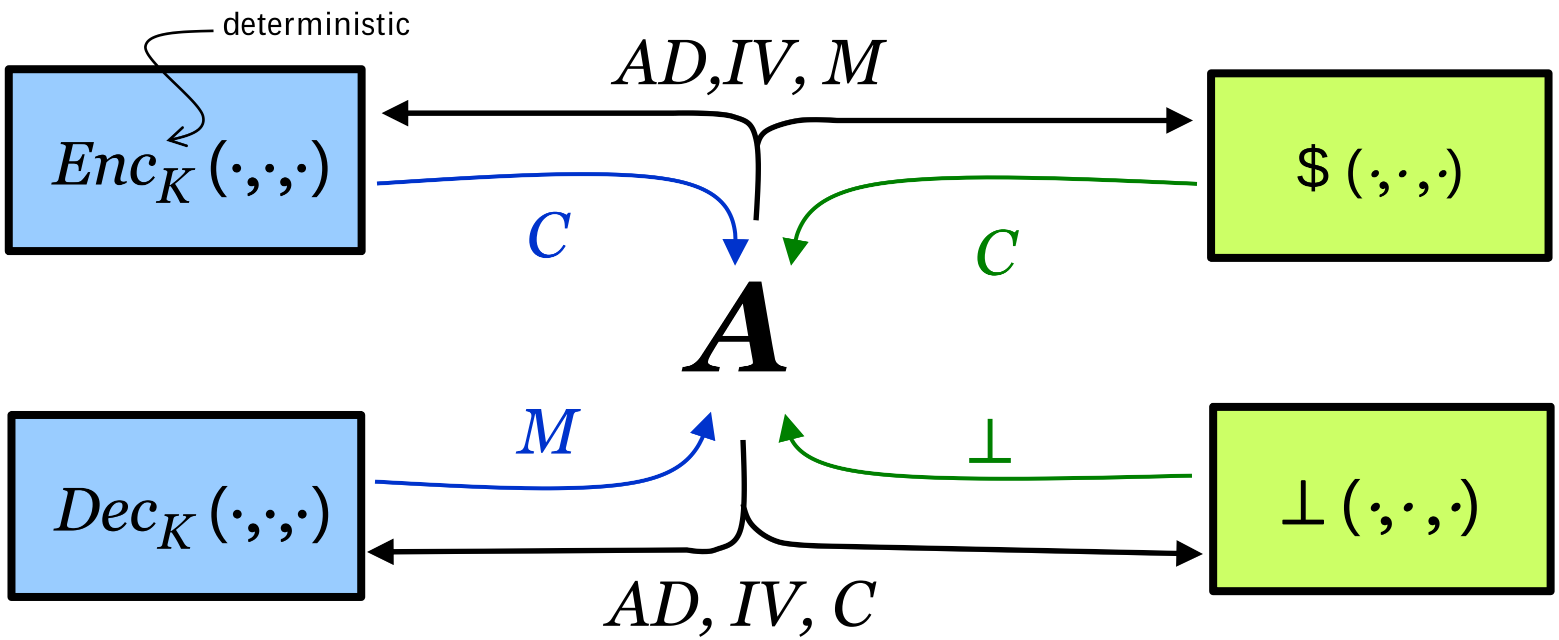
MRAE

Nonce/IV misuse – repeating an old value

- If the IV is a **nonce**, you get what an AE delivers
- If the IV gets **reused**, all that leaks is repetitions
  - authenticity is undamaged
  - privacy damaged to the extent unavoidable—repetitions of (AD, M) revealed

Misuse-Resistant AE

MRAE

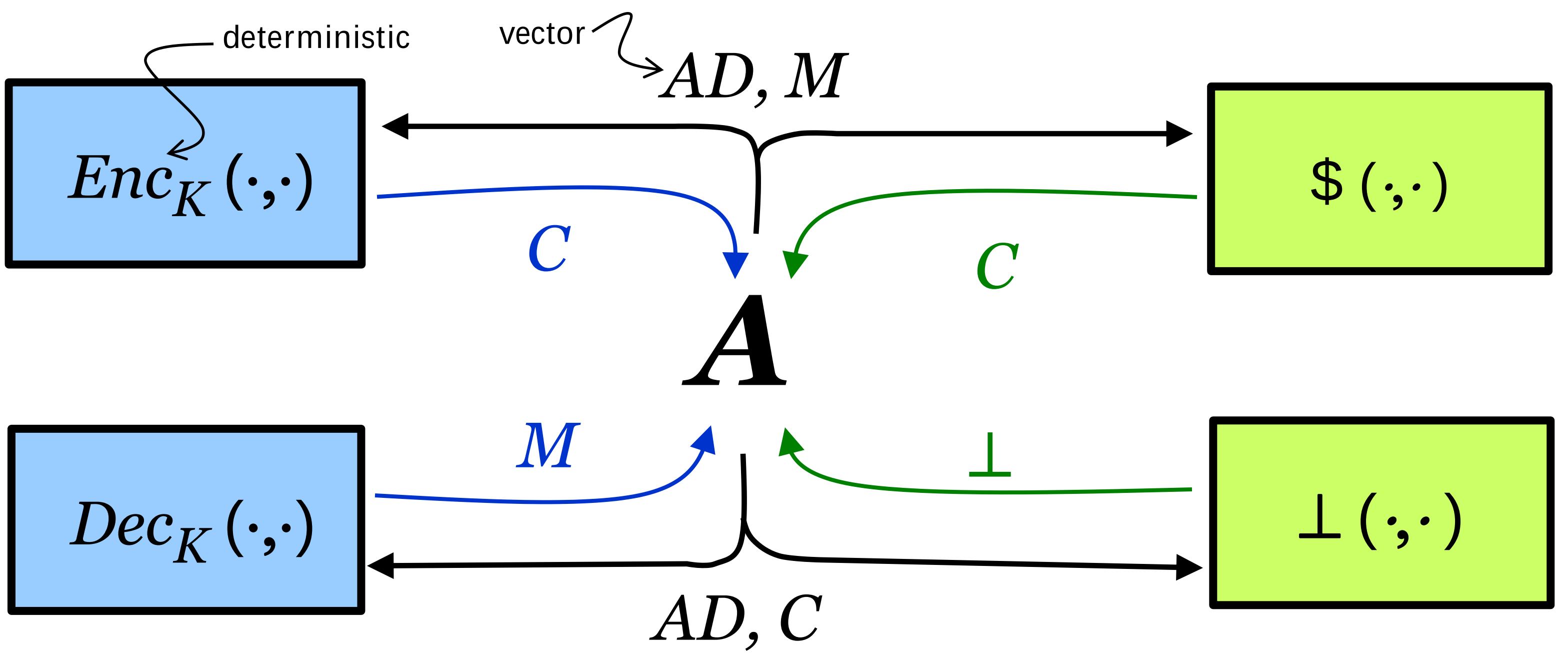


A may not: ask an enc query (AD, M, C);
ask a dec query of (AD,IV, C) after an enc query (AD, IV, M) returns C

Deterministic AE

(≈ "PRI" – "pseudorandom injection") [R, Shrimpton 2006]

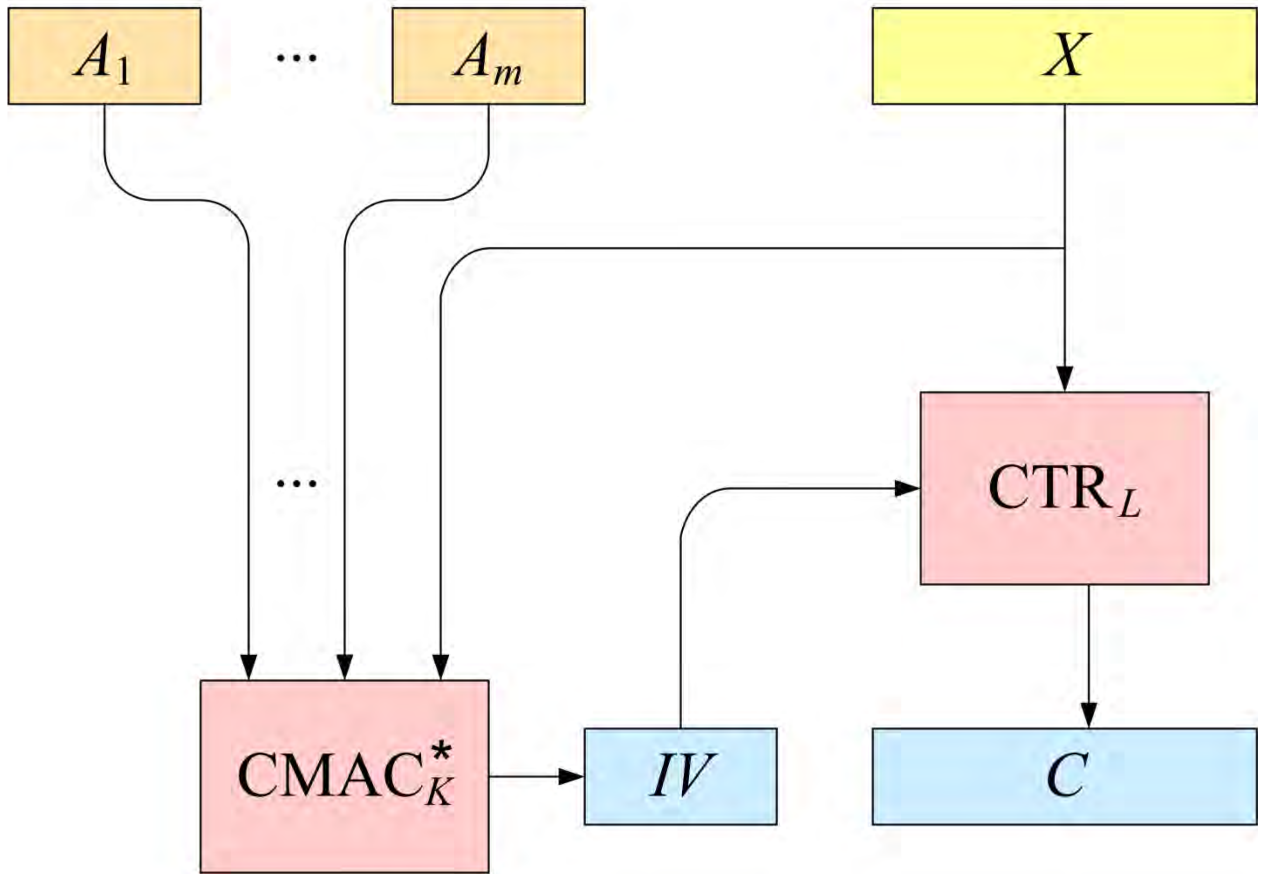
DAE



$\boldsymbol{A}$ may not: ask a dec query (AD, C) after an enc query (AD, M) returns C;
repeat a query

DAE ➔ MRAE: Regard one component of the AD as the IV

SIV



ISO/IEC 19772:2009
RFC 5297

A Traditional Taxonomy

Confusion/diffusion: one atomic primitive

- **Helix**, **SOBER**, ...

Composed: ind$-secure symmetric encryption + PRF

"two-pass"

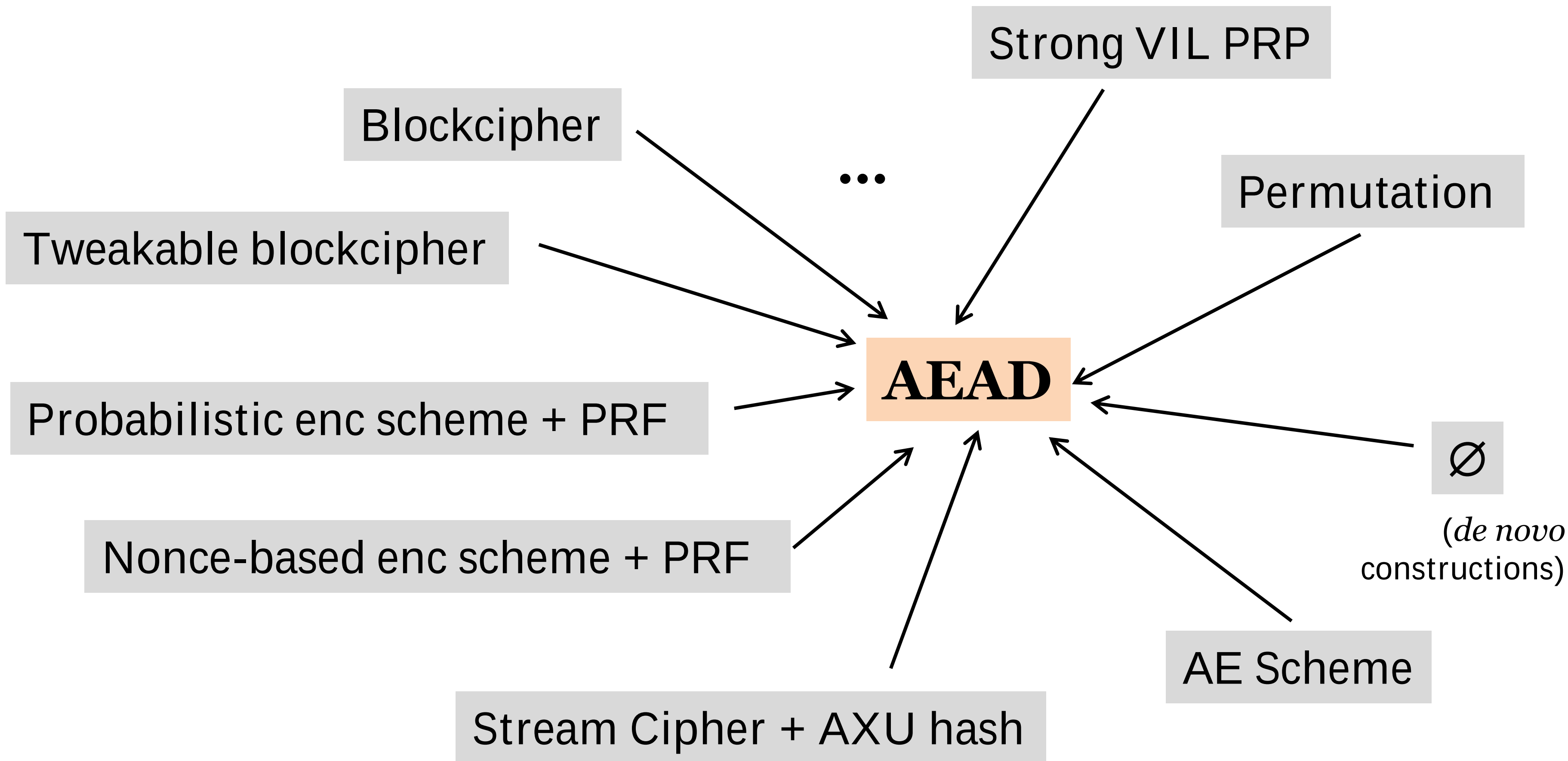
- **EtM**
- **CCM, GCM, etc.**

Integrated: blend privacy/authenticity parts

"one-pass"

- **OCB**

A Different Taxonomy



Patents

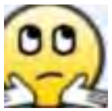
Initial filings in 2000
Gligor and Donescu (1/31)
Jutla(4/14)
Rogaway (10/12)

- **6,973,187** Gligor and Donescu 2005.12.06
- **6,963,976** Jutla 2005.11.07
- **7,093,126** Jutla 2006.08.15
- **8,107,620** Julta 2012.01.31
- **7,046,802** Rogaway 2006.05.16
- **7,200,227** Rogaway 2007.04.03
- **7,949,129** Rogaway 2011.05.24

- **7,840,003**. Kim, Han, Yoo, and Kwon. High-speed GCM-AES block cipher apparatus and method. Nov. 2010.
- **7,853,801**. Kim, Kwon, and Kim. System and method for providing authenticated encryption in GPON network [sic]. December 2010.
- **7,970,130**. Yen. Low-latency method and apparatus of GHASH operation for authenticated encryption Galois Counter Mode [sic]. June 28, 2011.
- **20080240423**. Gueron. Speeding up Galois Counter Mode (GCM) computations.
- **20060126835**. Chen and Buckingham. Authenticated encryption method and apparatus.
- **20090310775**. Gueron and Kounavis. Using a single instruction multiple data (SIMD) instruction to speed up Galois Counter Mode (GCM) computation.

⋮

Do Gligor or Jutla Patents Read Against OCB?



I don’t know.

1. An encryption method for providing both data confidentiality and integrity for a message, comprising the steps of:

receiving an input plaintext string comprising a message and padding it as necessary such that its length is a multiple of l bits;

partitioning the input plaintext string a length that is a multiple of l bits into a plurality of equal-size blocks of l bits in length;

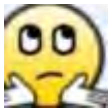
creating an a Manipulation Detection Code (MDC) block of l bits in length that includes the result of applying a non-cryptographic MDC function to the plurality of the equal-size blocks;

making one and only one processing pass with a single cryptographic primitive over each of said equal-size blocks and the MDC block to create a plurality of hidden ciphertext blocks each of l bits in length; and

performing a randomization function over said plurality of hidden ciphertext blocks to create a plurality of output ciphertext blocks each of l bits in length.

Claim 1 of Gligor-Donescu #6,973,187

Do Gligor or Jutla Patents Read Against OCB?



I don't know.

Claim 1 of Jutla's #8,107,620

1. A method for encrypting a sequence of plain-text messages using an n-bit block-cipher, the method comprising:
   choosing first and second secret keys;
   initializing an initial vector;
   initializing a pair-wise differentially uniform sequence generator using the said second secret key and the said initial vector;
   inputting at least one of a plurality of plain-text messages into an encryptor comprising a series of cipher blocks;
   generating a sequence of pair-wise differentially uniform random numbers using the said pair-wise differentially uniform sequence generator;
   updating the pair-wise differentially uniform sequence generator;
   updating the said initial vector;
   processing said at least one of a plurality of plain-text messages, and the said initial vector, and the said pair-wise differentially uniform random numbers, and the said first secret key, in the said encryptor to produce at least one of a plurality of encrypted cipher-text messages with embedded message integrity check, including separating said one of the plain-text messages into a plurality of plain-text blocks, combining each of the plain-text blocks with a respective one of said differentially uniform random numbers to generate a plurality of resultant text blocks and passing the plurality of resultant text blocks concurrently, in parallel through the series of cipher blocks, including passing each of the resultant text blocks through a respective one of the cipher blocks to produce said at least one of a plurality of encrypted cipher-text messages with embedded message integrity check in a single pass of the one of the plain-text messages through said series of cipher blocks; and
   using one or more processing units, executing an encryption program, to perform said processing.

For my part …

Announcement

I plan to freely license anything
- open-source
- software – except to the military
- research or non-commercial

The above is not a license. The actual license, in proper legal language, will be dropped to the web in the coming days.

I have the **draft language** with me, thanks to Harvard's Cyberlaw Clinic at the Berkman Center for Internet and Society

This is a **request for comments** on the draft.

Suggestion #1

Don't underestimate the value of theory for realizing fast and correct AE.

4 JULY 2001

DUAL COUNTER MODE

MIKE BOYLE

CHRIS SALTER

INTRODUCTION

For the past 18 months, the NSA has been developing a high-speed encryption mode for IP packets. The mode that we designed is identical in many aspects to Jutla's Integrity Aware Parallelizable Mode (IAPM). There is one important difference in our proposal. In the IP world, a large number of packets might arrive out of order. Integrity Aware Parallelizable Mode (IAPM) and the proposed variations incur a large overhead for out of order packets[JU 01]. Each packet requires at least the time to perform a full decryption to obtain an IV before decryption of the cipher can begin. This note describes our solution to this problem.

First, we describe the basic mode and its features. We then describe how to implement this mode for IPSec.

DUAL COUNTER MODE

Dual counter mode is a hybrid of ECB mode and counter mode. Let E represent encryption by a codebook of width W. Let $P_1, P_2, \ldots, P_j$ be j blocks of plaintext and let $C_1, C_2, \ldots, C_j$ be the corresponding ciphertext. Let f be a polynomial of degree W for a primitive linear feedback shift register. Also, let $\{x_i\}$ be the sequence of fills generated by this polynomial. The first fill, x_0, is a secret shared between the two peers. This initial fill is most easily derived from the key exchange[1]. Dual counter mode can be described as follows:

j = # of datablocks

Broken within days
by Rogaway and
by Donescu, Gligor,
and Wagner

NATIONAL SECURITY AGENCY CENTRAL SECURITY SERVICE

Defending Our Nation. Securing The Future.

HOME | ABOUT NSA | ACADEMIA | BUSINESS | CAREERS | INFORMATION ASSURANCE | RESEARCH | PUBLIC INFORMATION | COMMITMENT

Research

- Security Enhanced Linux
- Information Assurance Research
- Mathematical Sciences Program
- Computer & Information Sciences Research
- Technology Transfer
  - Advanced Computing
  - Advanced Mathematics
  - Communications & Networking
  - Information Processing
  - Microelectronics
  - Other Technologies
- Technology Fact Sheets
- Publications
- Related Links

Home > Research > Technology Transfer > Technology Fact Sheets

SEARCH

Dual Counter Mode: A new mode for pipelined encryption and data integrity

Aliases:
Dual Counter Mode

Technical Challenge:
There are several modes of encryption in use today, but none of them combine the ability to pipeline encryption with the ability to use a simple (essentially pipelined) checksum for data integrity.

Description:
Dual Counter Mode is a codebook encryption mode that combines features of several previous modes to get the advantages of each while eliminating their flaws. The mode provides confidentiality and data integrity without sacrificing pipelining. It also overcomes the problem with out of order packets.

Demonstration Capability:
A demonstration is not available.

Potential Commercial Application(s):
This technology is a solution to the problem of encrypting and authenticating IPSec data between high-speed gateways and backbone routers.

Patent Status:
A patent application has been filed with USPTO.

Reference Number: 1308

If you are interested in exploring this technology further, please call express your interest in writing to the:

National Security Agency
NSA Technology Transfer Program
9800 Savage Road, Suite 6541
Fort George G. Meade, Maryland 20755-6541

Date Posted: Jan 15, 2009 | Last Modified: Jan 15, 2009 | Last Reviewed: Jan 15 2009

Defending Our Nation. Securing The Future.

2009 National Security Agency ★ Privacy | Terms of Use | NoFEAR Act | FOIA | DNI.gov | DOD.mil | USA.gov | Contact Us | Site Map

Ironic Follow-Up

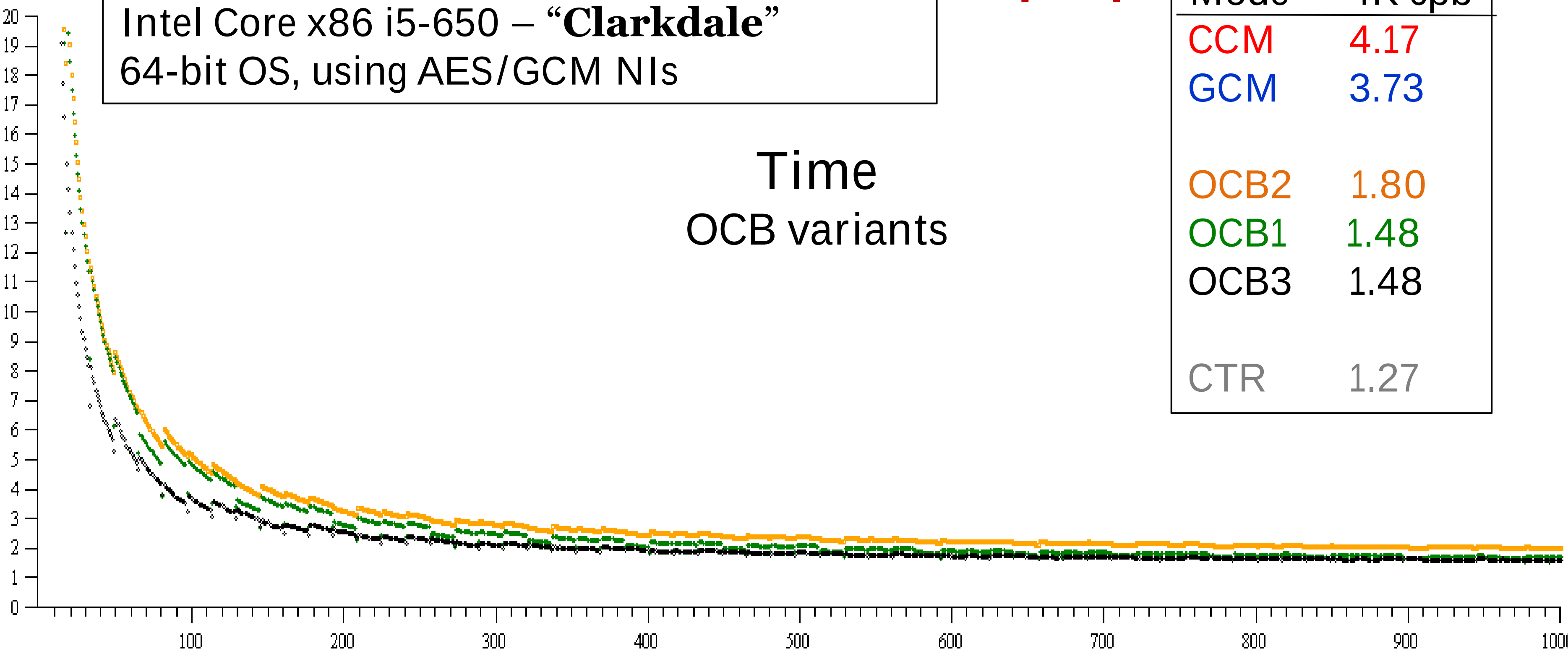
http://www.nsa.gov/research/tech_transfer/fact_sheets/dual_counter_mode.shtml

Suggestion #2

Don't underestimate the value of implementation for understanding what's fast or practically desirable.

Software Performance
Intel Core x86 i5-650 – "**Clarkdale**"
64-bit OS, using AES/GCM NIs

[KR11]

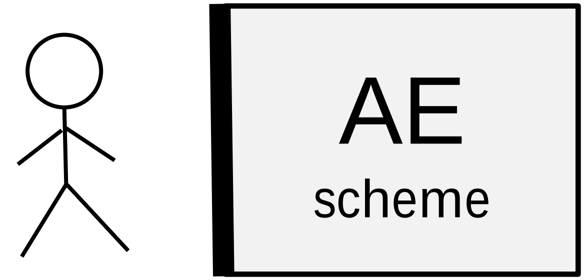


Mode	4K cpb
CCM	4.17
GCM	3.73
OCB2	1.80
OCB1	1.48
OCB3	1.48
CTR	1.27

Suggestion #3

Mind the API

and make it incremental




```
int ae_encrypt(
  ae_ctx      *ctx,
  const void *nonce,
  const void *pt,
  int          pt_len,
  const void *ad,
  int          ad_len,
  void        *ct,
  void        *tag,
  int          final);
```

If nonce!=NULL then a message is being initiated. If final!=0 then a message is being finalized. If final==0 or nonce==NULL then the incremental interface is being used. If nonce!=NULL and ad_len<0, then use same ad as last message.
Returns: if nonnegative, the number of bytes written to ct ...

Challenge the atomicity of plaintexts, ciphertexts in defs and schemes

[Boldyreva, Degabriele, Paterson, Stam 2012]

```
int ae_decrypt(
  ae_ctx      *ctx,
  const void *nonce,
  const void *ct,
  int          ct_len,
  const void *ad,
  int          ad_len,
  void        *pt,
  const void *tag,
  int          final);
```

If nonce!=NULL then "ct" points to the start of a ciphertext. If final!=0 then "in" points to the final piece of ciphertext. If final==0 or nonce== NULL then the incremental interface is being used. If nonce!=NULL and ad_len<0, then use same ad as last message.

The main part of the API from Krovetz's implementation of OCB3

Suggestion #4

Standardize

- A **few** schemes, of different types or characteristic
- The **best** schemes, irrespective of patents

ISO/IEC 19772:2009

1. CCM
2. EAX
3. GCM
3. SIV
5. OCB2
6. EtM

Suggestion #5

Recognize

The myth of requirements

English: what is *necessary*
Industry: what *someone* wants

Tradeoffs – not requirements
Eg:
- single vs. multiple underlying keys
- vector-valued AD vs. string-valued
- MRAE vs. standard AE vs. online DAE
- BB security vs. something beyond

...

A Few Research Questions

1. Can beyond-birthday-bound security be achieved **cheaply** and **generically**

2. Better **definitions**, and **constructions**, for **online** MRAE (memory usage a parameter)

3. Less atomic, more **API-centric** definitions and constructions

4. A theory useful for making **stream ciphers** into AE schemes with added cost $\ll$ universal hashing