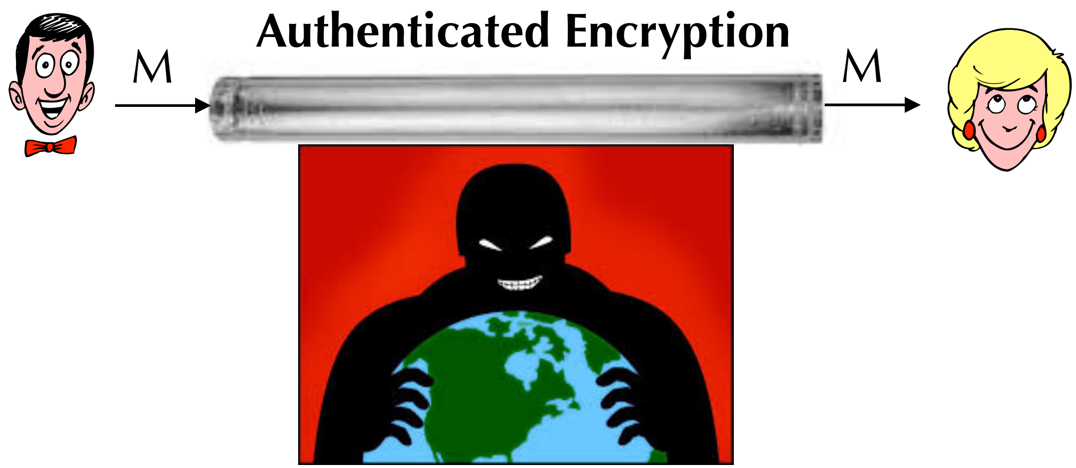
(A brief, incomplete)

Introduction to Authenticated Encryption

Tom Shrimpton

Summer School on Real-World Crypto and Privacy

June 11, 2018


Authenticated Encryption
M
M

It's complicated...

Probabilistic or deterministic AE?

Nonce based AE?

What happens if a nonce repeats?

Do I need to support associated data?

What primitives should we build upon?

encryption + MAC? (tweakable) wide-block ci
sponges? ...

What should happen when decryption fails?

Is it safe to provide multiple, descriptive exceptions/error messages?

Stop all future processing, or just for this message?

What kind of information can decryption safely leak?

Safe to release plaintext data "early"?

Online encryption/decryption property?

"Atomic" plaintexts/ciphertexts, or stream-based?

(Authenticated encryption != Secure Channel)

Let's start at the beginning: syntax

An Encryption Scheme is a triple of algorithms $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$

Key-generation algorithm $\mathcal{K}$ samples from a specified key space

Encryption algorithm $\mathcal{E} : \mathcal{K} \times \{0,1\}^* \to \{0,1\}^* \cup \{\perp\}$

Classically, randomized or stateful
$C \xleftarrow{\$} \mathcal{E}_K(M)$

Decryption algorithm $\mathcal{D} : \mathcal{K} \times \{0,1\}^* \to \{0,1\}^* \cup \{\perp\}$

Always deterministic
$M \leftarrow \mathcal{D}_K(C)$

Correctness:

$$\forall K \in \mathcal{K}, \forall M \in \{0,1\}^* : \; \Pr\Big[C \xleftarrow{\$} \mathcal{E}_K(M) : C = \perp \text{ or } \mathcal{D}_K(C) = M \Big] = 1$$

(note: logically equivalent to $C \neq \perp \implies \mathcal{D}_K(C) = M$)

Privacy: Indistinguishability from random bits (IND$-CPA)

$\underline{\mathbf{Exp}_{\Pi}^{\text{ind\$-cpa}}(A)\text{:}}$

$K \xleftarrow{\$} \mathcal{K}$

$d \xleftarrow{\$} \{0,1\}$

$d' \xleftarrow{\$} A^{\mathcal{O}(\cdot)}$

If $d' = d$ then Return 1

Return 0

$\underline{\textbf{Oracle } \mathcal{O}(M)}$

$Y_1 \xleftarrow{\$} \mathcal{E}_K(M)$

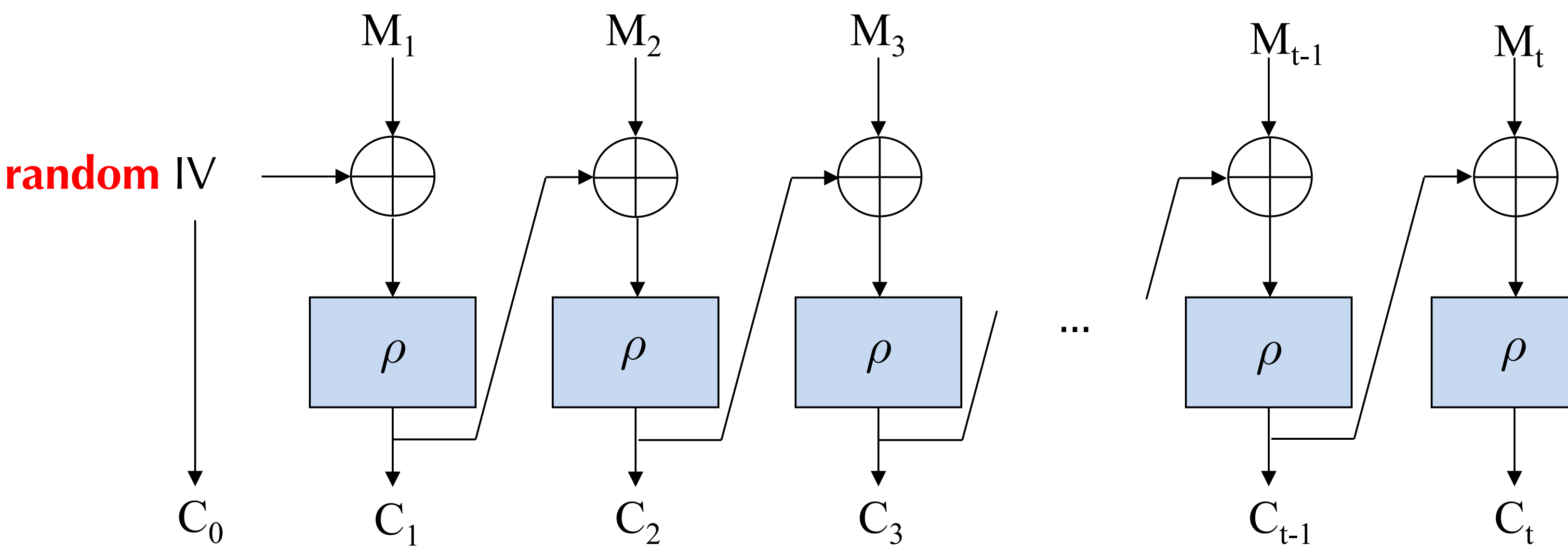
$Y_0 \xleftarrow{\$} \{0,1\}^{|Y_1|}$

Return Y_d

$$\mathbf{Adv}_{\Pi}^{\text{ind\$-cpa}}(A) = 2\Pr\left[\mathbf{Exp}_{\Pi}^{\text{ind\$-cpa}}(A) = 1\right] - 1$$

CBC-mode encryption is IND\$-CPA secure

(If M isn't block aligned, then return $\perp$)

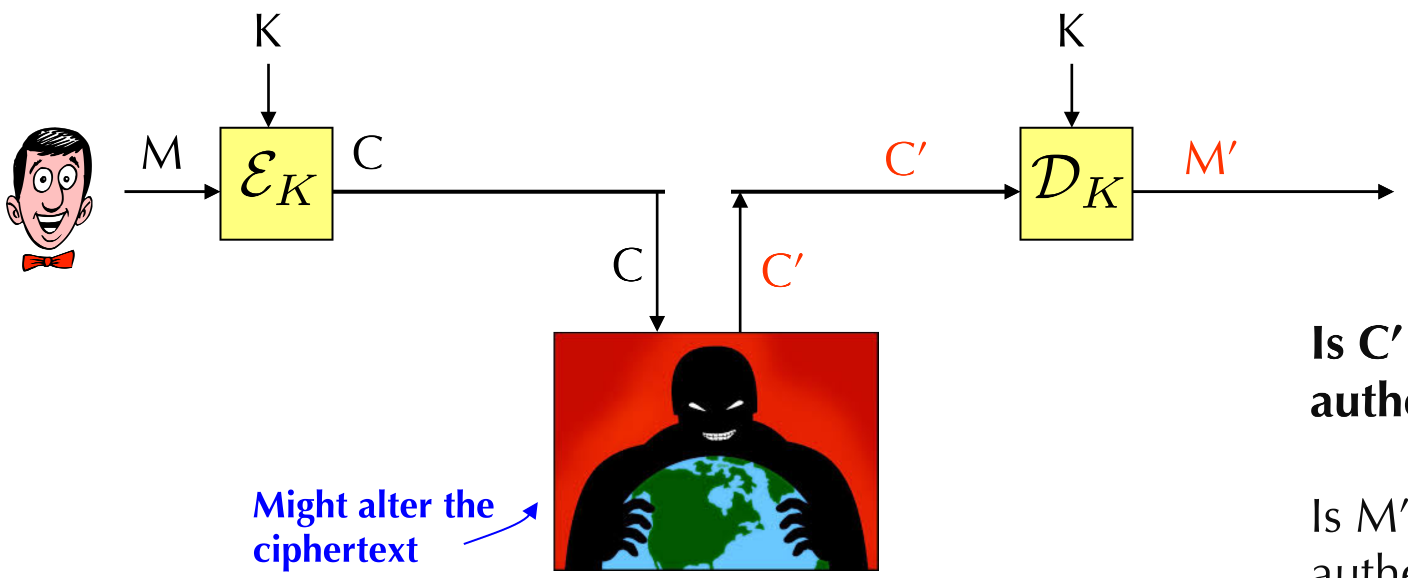


If $\rho \xleftarrow{\$} \mathsf{Func}(n, n)$ then **all output blocks random**, up to a birthday-bound term.

(the set of all functions $f: \{0,1\}^n \to \{0,1\}^n$)

Privacy? ✓ What about authenticity?

Alice wants to be sure that what she receives is what was sent.



Is C′ an authentic ctxt?

Is M′ an authentic ptxt?

Authenticity: Integrity of Ciphertexts (INT-CTXT)

(Bellare, Rogaway AC' 00) (Katz, Yung FSE' 00) (Bellare, Namprempre AC'00)

$\underline{\mathbf{Exp}_{\Pi}^{\text{int-ctxt}}(A)\text{:}}$

$K \xleftarrow{\$} \mathcal{K}$

$b \xleftarrow{\$} \{0,1\}$

$b' \xleftarrow{\$} A^{\mathcal{E}_K(\cdot),\mathcal{O}(\cdot)}$

If $b' = b$ then Return 1

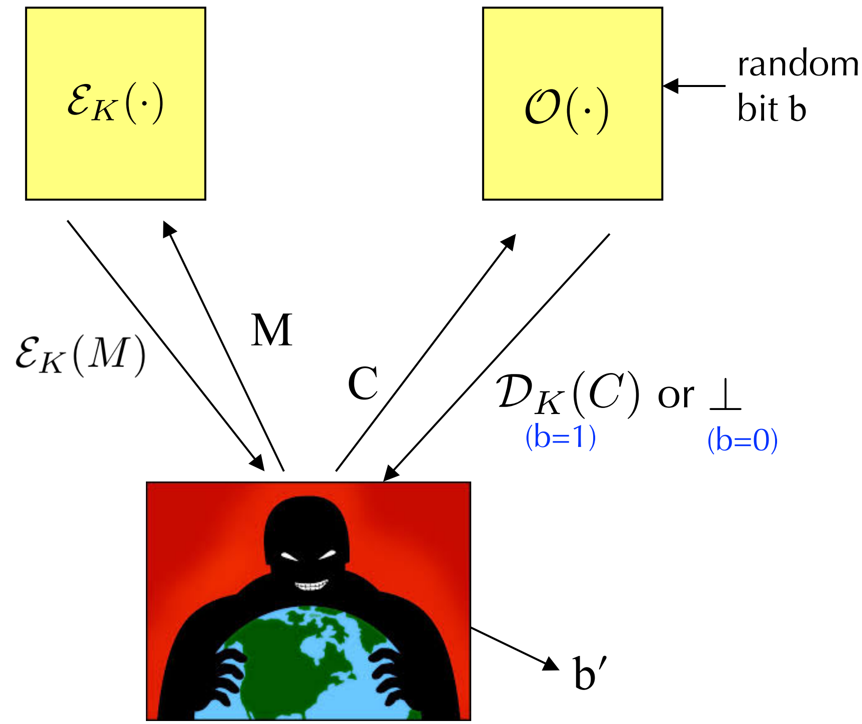
Return 0

$\underline{\text{Oracle } \mathcal{O}(C)\text{:}}$

If $b = 0$ then Return $\perp$

Return $\mathcal{D}_K(C)$

$$\mathbf{Adv}_{\Pi}^{\text{int-ctxt}}(A) = 2\Pr(\mathbf{Exp}_{\Pi}^{\text{int-ctxt}}(A) = 1) - 1$$



To prevent "trivial wins" of the game, adversary is forbidden to ask C of the right oracle if C was returned by the left oracle

Definition of AE security (informally)

If encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is
IND$-CPA secure **and** INT-CTXT secure then it is "AE secure".

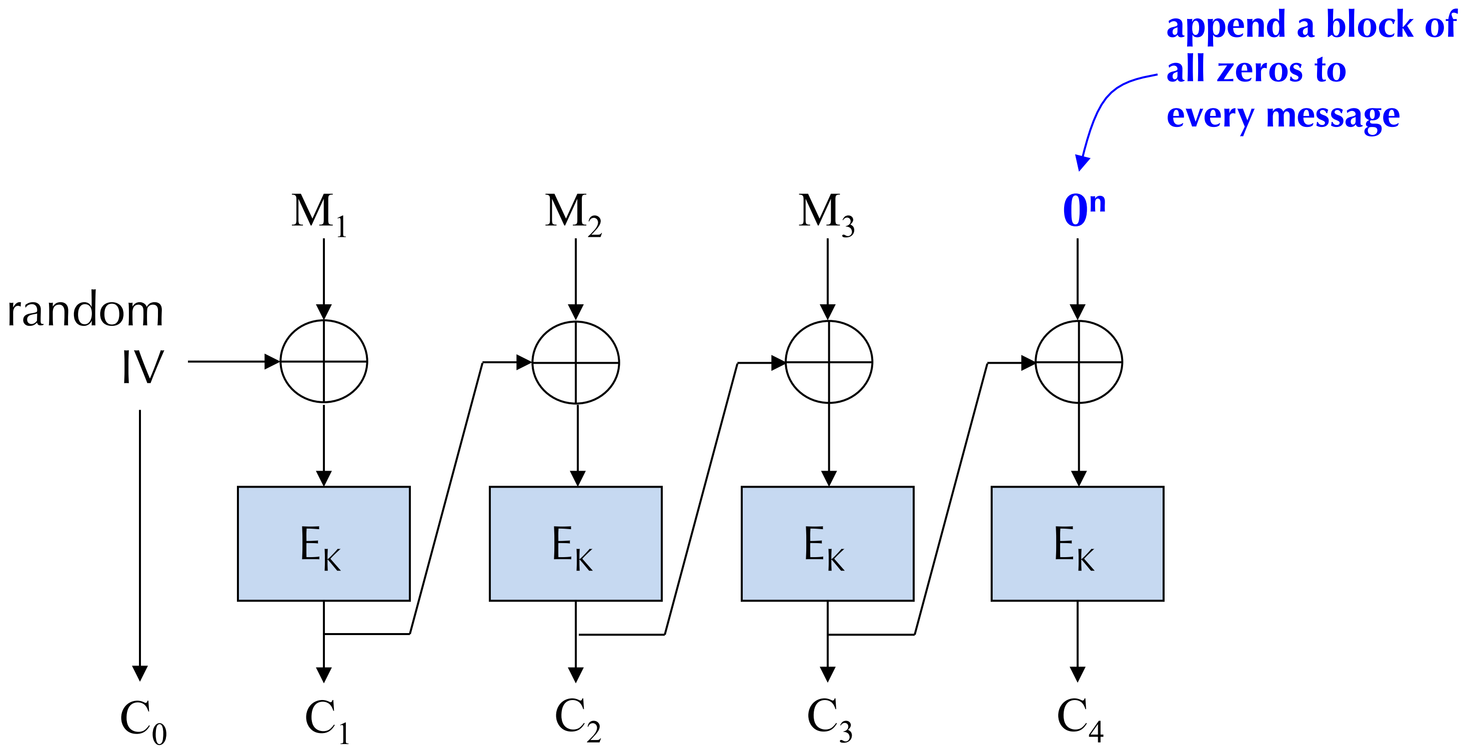
Definition of AE security (informally)

If encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is
IND$-CPA secure **and** INT-CTXT secure then it is "AE secure".

ctxts look like
random bitstrings

dishonestly
created ctxts
decrypt to $\perp$

Folklore idea: add "redundancy" to encryption

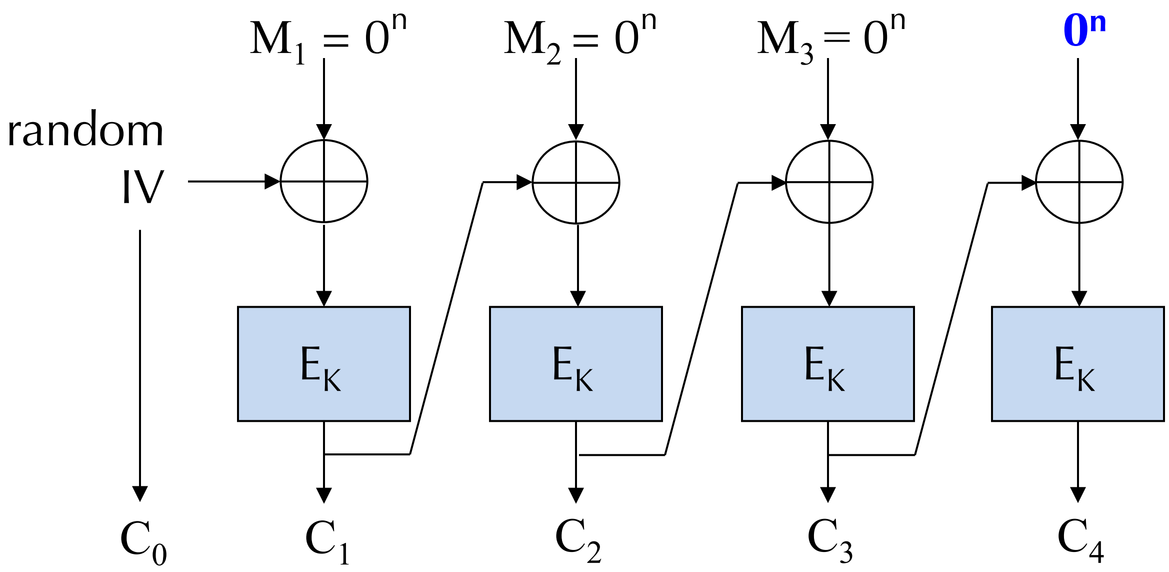


Decryption: If last block is all zeros, then return $M_1\, M_2\, M_3$

Else return $\perp$

Folklore idea: add "redundancy" to encryption

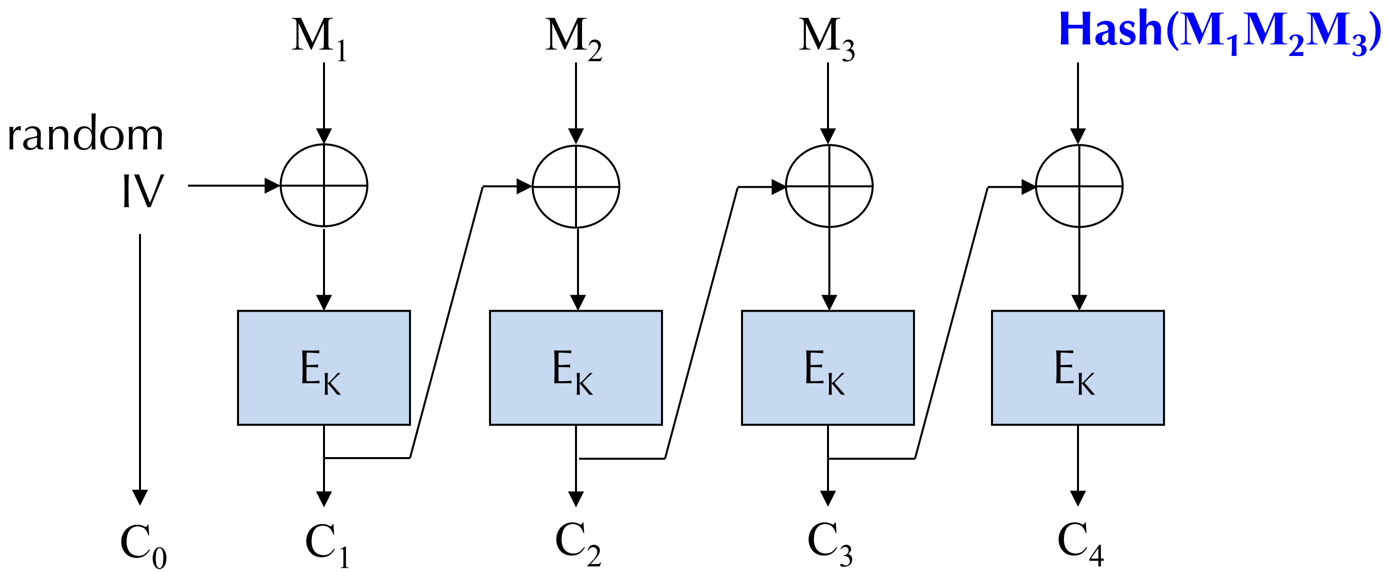
Doesn't work!



Easy to "forge" a dishonest ciphertext that is valid.

Folklore idea: add "redundancy" to encryption

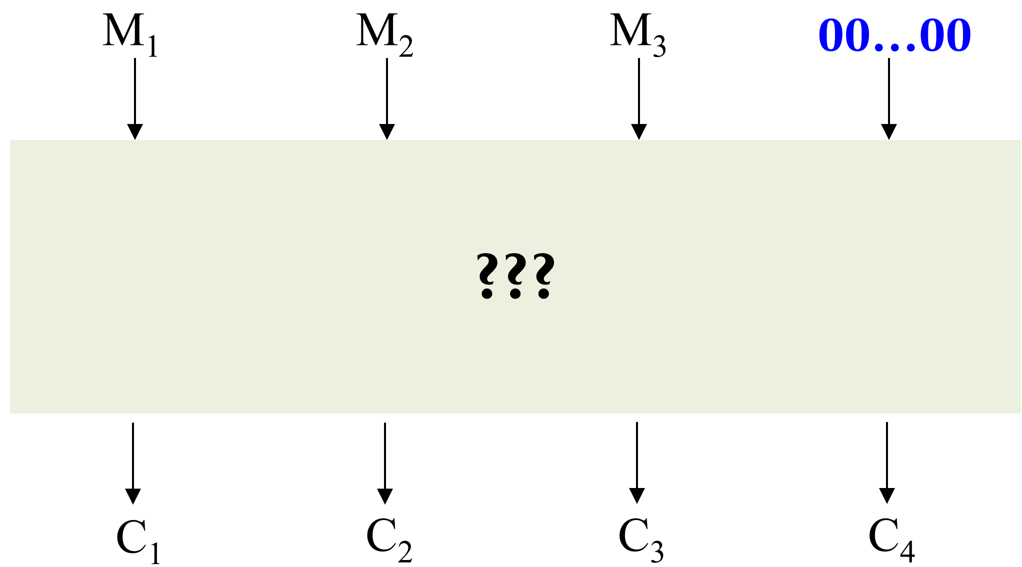
Doesn't work!



In fact, **no** publicly computable redundancy works.

(An, Bellare EC'01)

Adding public redundancy *can* be made to work… (more later)



AE via "generic composition" of encryption and MAC

SSH: $\mathcal{E}_{K_1,K_2}(M) = \overline{\mathcal{E}}_{K_1}(M) || F_{K_2}(M)$ **"Encrypt and MAC"**

SSL/TLS: $\mathcal{E}_{K_1,K_2}(M) = \overline{\mathcal{E}}_{K_1}(M || F_{K_2}(M))$ **"MAC then Encrypt"**

IPSec: $\mathcal{E}_{K_1,K_2}(M) = \overline{\mathcal{E}}_{K_1}(M) || F_{K_2}(\overline{\mathcal{E}}_{K_1}(M))$ **"Encrypt then MAC"**

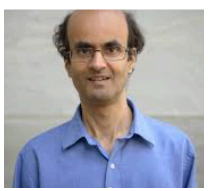
AE via "generic composition" of encryption and MAC

SSH: $\mathcal{E}_{K_1,K_2}(M) = \overline{\mathcal{E}}_{K_1}(M)||F_{K_2}(M)$ **"Encrypt and MAC"**

SSL/TLS: $\mathcal{E}_{K_1,K_2}(M) = \overline{\mathcal{E}}_{K_1}(M||F_{K_2}(M))$ **"MAC then Encrypt"**

IPSec: $\mathcal{E}_{K_1,K_2}(M) = \overline{\mathcal{E}}_{K_1}(M)||F_{K_2}(\overline{\mathcal{E}}_{K_1}(M))$ **"Encrypt then MAC"**

(Bellare, Namprempre AC' 00)



"Which of these is AE-secure given
any secure encryption scheme $\overline{\Pi} = (\overline{\mathcal{K}}, \overline{\mathcal{E}}, \overline{\mathcal{D}})$ and
any secure MAC F ?" (paraphrasing)

AE via "generic composition" of encryption and MAC

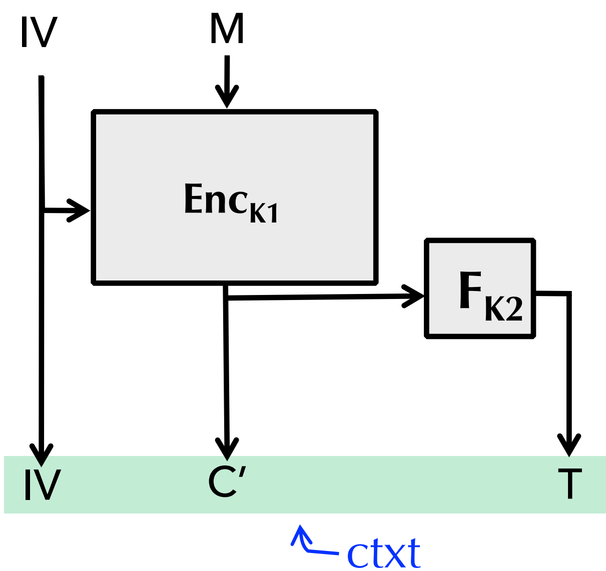
SSH:	$\mathcal{E}_{K_1,K_2}(M) = \overline{\mathcal{E}}_{K_1}(M) \| F_{K_2}(M)$	**"Encrypt and MAC"**
SSL/TLS:	$\mathcal{E}_{K_1,K_2}(M) = \overline{\mathcal{E}}_{K_1}(M \| F_{K_2}(M))$	**"MAC then Encrypt"**
IPSec:	$\mathcal{E}_{K_1,K_2}(M) = \overline{\mathcal{E}}_{K_1}(M) \| F_{K_2}(\overline{\mathcal{E}}_{K_1}(M))$	**"Encrypt then MAC"**

(Bellare, Namprempre AC' 00)


"Which of these is AE-secure given
any secure encryption scheme $\overline{\Pi} = (\overline{\mathcal{K}}, \overline{\mathcal{E}}, \overline{\mathcal{D}})$ and
any secure MAC F ?" (paraphrasing)

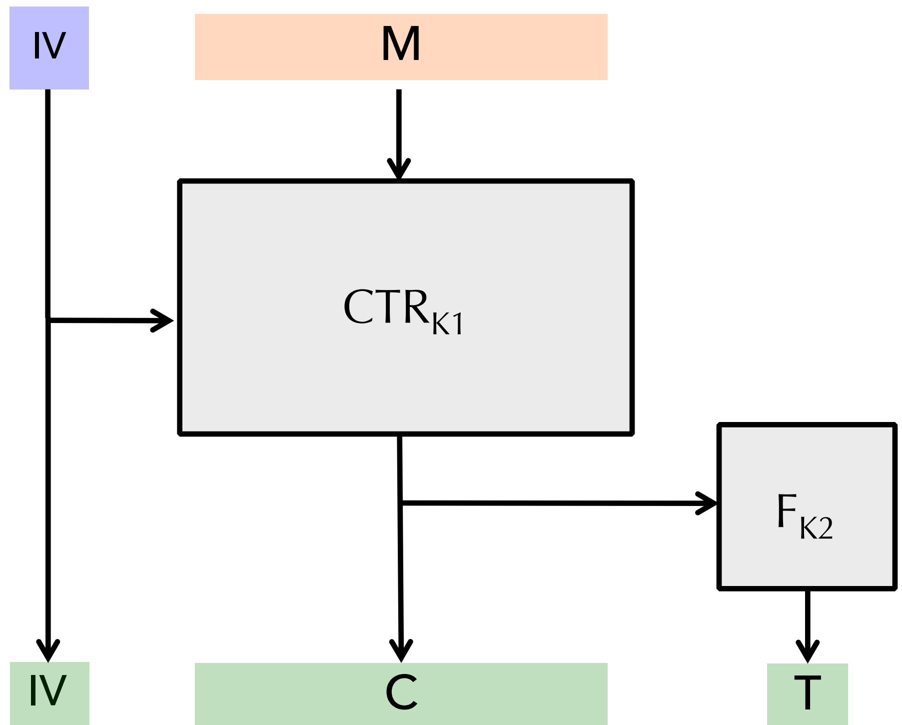
ISO/IEC 19772, Mechanism 5 (Encrypt-then-MAC)

Information Security – Security Techniques – Authenticated Encryption



"Enc" = CBC, CTR, OFB, CFB blockcipher modes

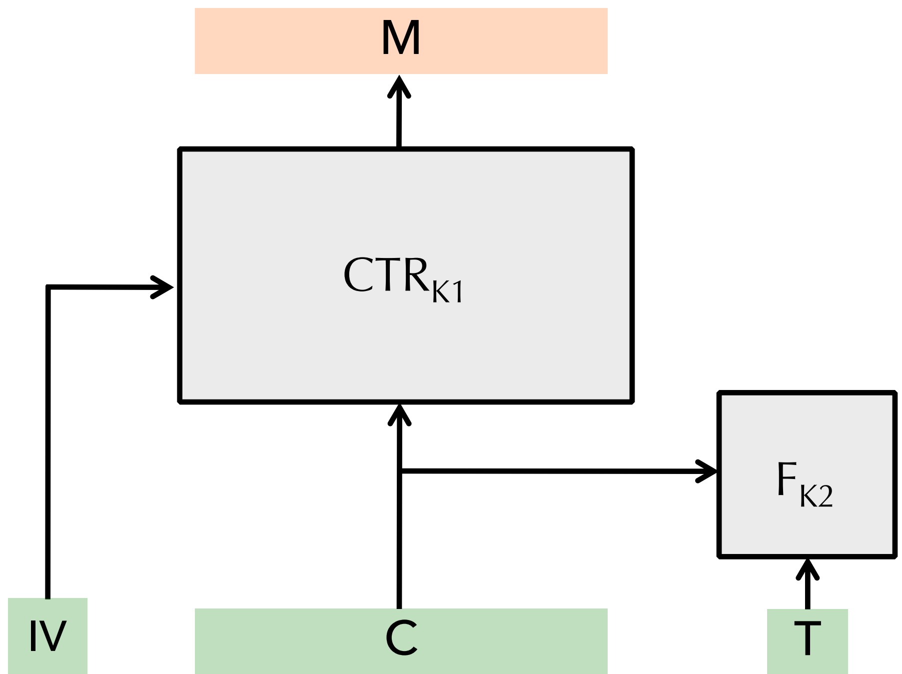
Encrypt-then-MAC over CTR mode (encryption)



IND$-CPA secure
encryption scheme

secure MAC
(e.g. PRF, like HMAC)

Encrypt-then-MAC over CTR mode (decryption)

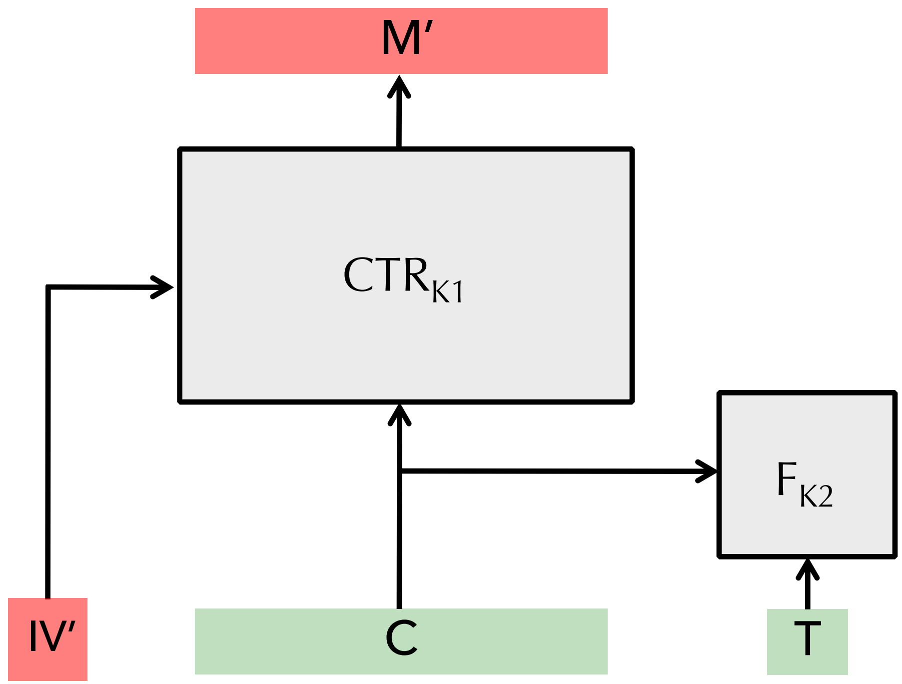


IND$-CPA secure
encryption scheme

secure MAC
(e.g. PRF, like HMAC)

if $F_{K2}(C) = T$ then return $Dec_{K1}(IV \| C)$
else return $\perp$

Encrypt-then-MAC over CTR mode (decryption)

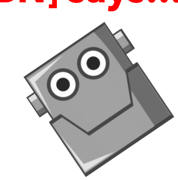


IND$-CPA secure
encryption scheme

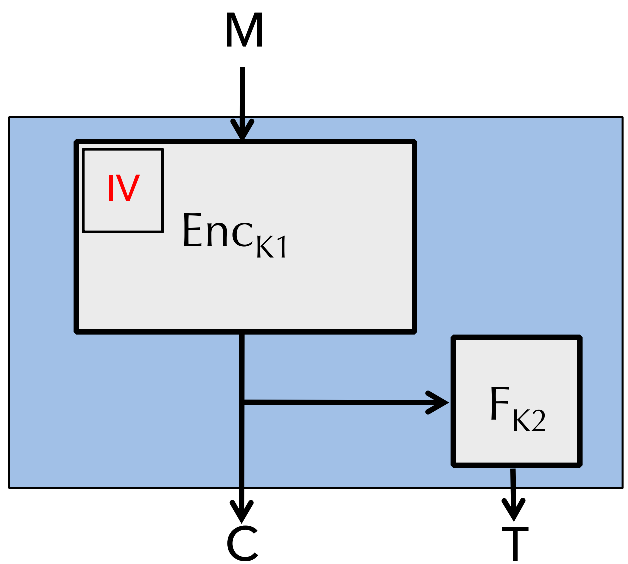
secure MAC
(e.g. PRF, like HMAC)

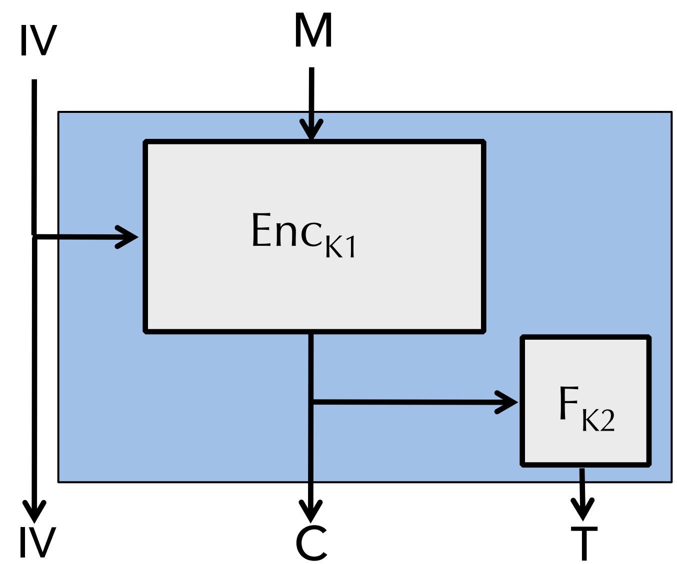
if $F_{K2}(C) = T$ then return $Dec_{K1}(IV \,||\, C)$
else return $\perp$

But… [BN] says… ???



[BN] Encrypt-then-MAC vs. ISO "Encrypt-then-MAC"





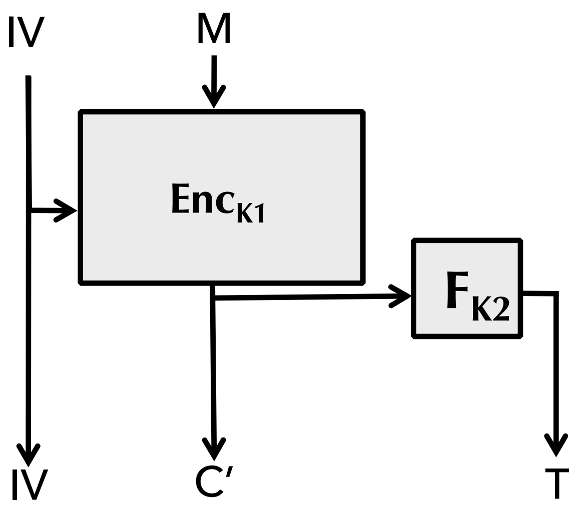
Probabilistic AE scheme built from a **probabilistic encryption** scheme and a MAC

[BN] is about this setting only.

Determinisitc AE scheme with an *explicitly surfaced IV* built from a **deterministic encryption** scheme with an *explicitly surfaced IV* and a MAC

Incorrect understanding of [BN], in practice

ISO/IEC 19772, Mechanism 5 (Encrypt-then-MAC)
Information Security – Security Techniques – Authenticated Encryption



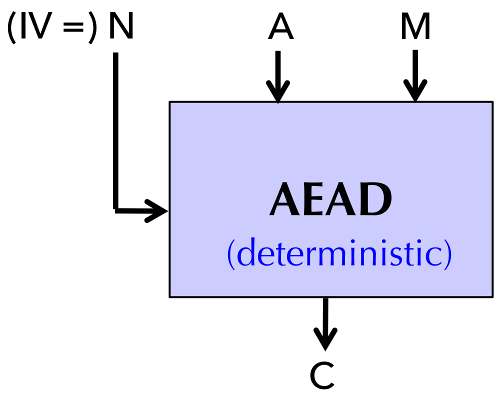
IV required to be a nonce (but not random)

"Enc" = CBC, CTR, OFB, CFB blockcipher modes
-- some require IV to be random for IND-CPA
-- not all have {0,1}* domains

IV not covered by tag

All are deviations from what [BN] analyzed.
Standard appeals to [BN] to justify security.

1. Typical goal nowadays is **nonce-based AE,** not probabilistic AE.
Moreover, the AE scheme should support **associated data.**

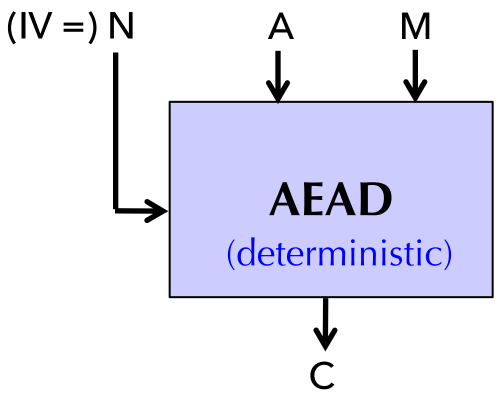


N= nonce (“number used once”, e.g. sequence number)

A = associated data, **bound to plaintext/ciphertext**, **not private**

M = plaintext, private

1. Typical goal nowadays is **nonce-based AE**, not probabilistic AE. Moreover, the AE scheme should support **associated data.**



N= nonce ("number used once", e.g. sequence number)

A = associated data, **bound to plaintext/ciphertext**, **not private**

M = plaintext, private

2. Standards and common crypto libraries **don't provide probabilistic encryption** schemes, they provide **deterministic encryption with an explicitly surfaced IV.**

```
int encrypt(unsigned char *plaintext,
            int plaintext_len,
            unsigned char *key,
            unsigned char *iv,
            unsigned char *ciphertext)
```

openSSL
encryption API

IV-based encryption with associated data

(Rogaway CCS'02)
(Rogaway, Namprempre, S. EC'14)

The IV space

(Deterministic!) **Encryption algorithm** $\mathcal{E}\colon (\mathcal{H} \times \mathcal{V}) \times \mathcal{K} \times \{0,1\}^* \to \{0,1\}^* \cup \{\perp\}$

The associated-data space, $\mathcal{H} = (\{0,1\}^*)^*$

Decryption algorithm $\mathcal{D}\colon (\mathcal{H} \times \mathcal{V}) \times \mathcal{K} \times \{0,1\}^* \to \{0,1\}^* \cup \{\perp\}$

All-in-one AEAD security notion

(Rogaway, S. EC'06)

$\underline{\mathbf{Exp}^{\mathrm{ae}}_{\Pi}(A)}$:

$K \xleftarrow{\$} \mathcal{K}$

$d \xleftarrow{\$} \{0,1\}$

$d' \xleftarrow{\$} A^{\mathsf{Enc}(\cdot,\cdot,\cdot),\mathsf{Dec}(\cdot,\cdot,\cdot)}$

If $d' = d$ then Return 1

Return 0

$\underline{\textbf{Oracle } \mathsf{Enc}(H, N, M)}$

$Y_1 \leftarrow \mathcal{E}_K^{H,N}(M)$

$Y_0 \xleftarrow{\$} \{0,1\}^{|Y_1|}$

Return Y_d

$\underline{\textbf{Oracle } \mathsf{Dec}(H, N, C)}$

$X_1 \leftarrow \mathcal{D}_K^{H,N}(C)$

$X_0 \leftarrow \bot$

Return X_d

$$\mathbf{Adv}^{\mathrm{ae}}_{\Pi}(A) = 2\Pr[\,\mathbf{Exp}^{\mathrm{ae}}_{\Pi}(A) = 1\,] - 1$$

An adversary that never repeats a nonce is called "**nonce-respecting**"

(Bellare, Namprempre AC' 00)


"Which of EaM, EtM, MtE gives a secure probabilistic AE scheme, given a secure probabilistic encryption scheme and a secure MAC?"

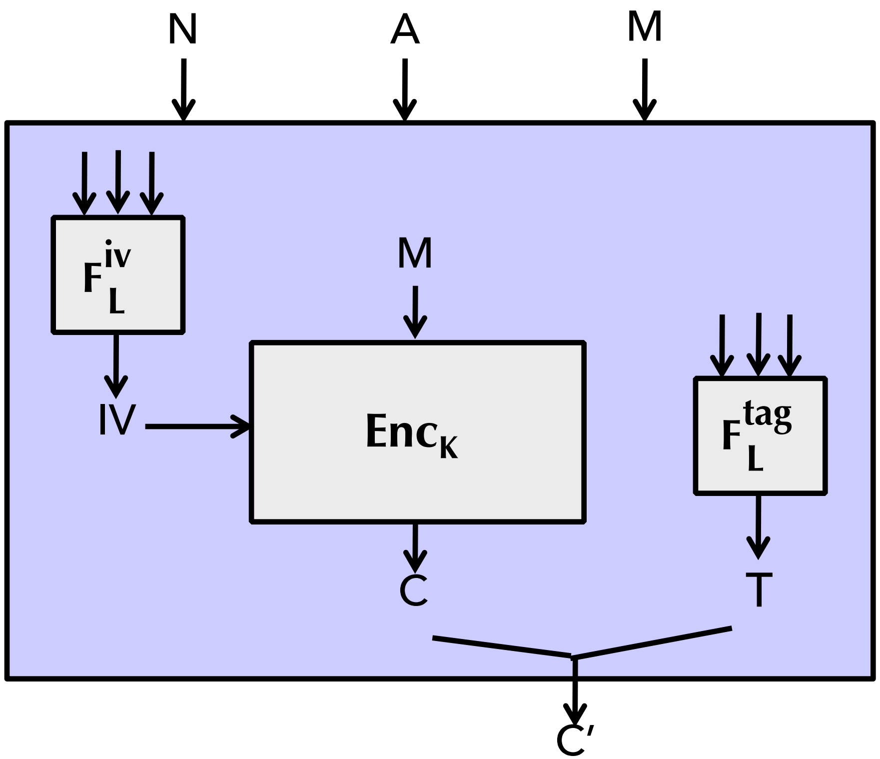
15 years!

(Namprempre, Rogaway, S. EC' 14)



"What are **all** of ways to build an IV-based AEAD scheme that is **secure with nonce IVs**, from a secure IV-based encryption scheme and a secure PRF?"

Our basic templates



F^{iv} inputs: (N or □ , A or □, M or □)

F^{tag} inputs: (N or □ , A or □, M or □) **"E&M"**

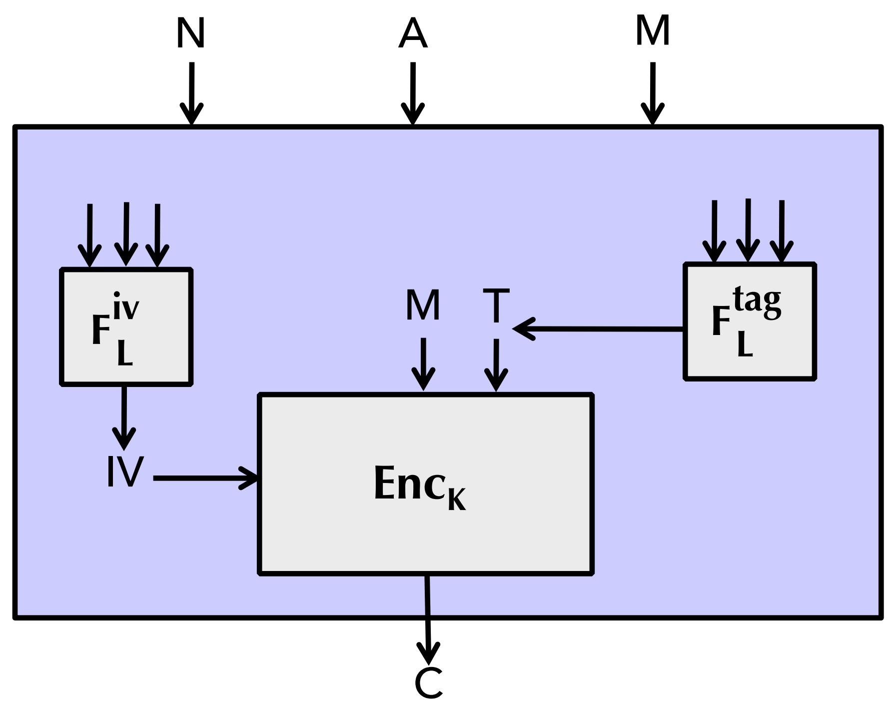
or (N or □ , A or □, C or □) **"EtM"**

□ = **"missing"**

F^{iv} inputs: (N or □ , A or □, M or □)

F^{tag} inputs: (N or □ , A or □, M or □)

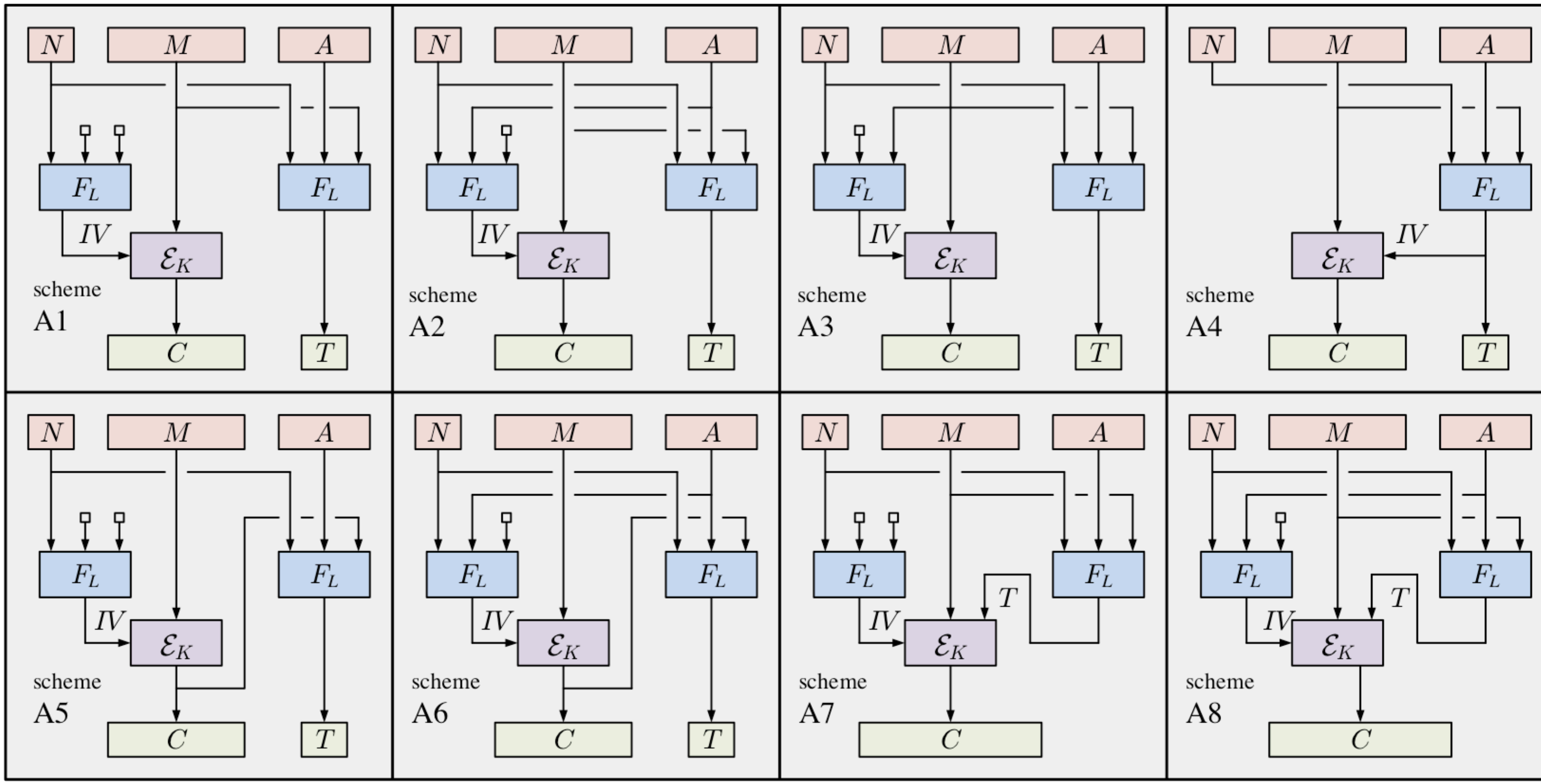
"MtE"



160 schemes to analyze!

The favored eight

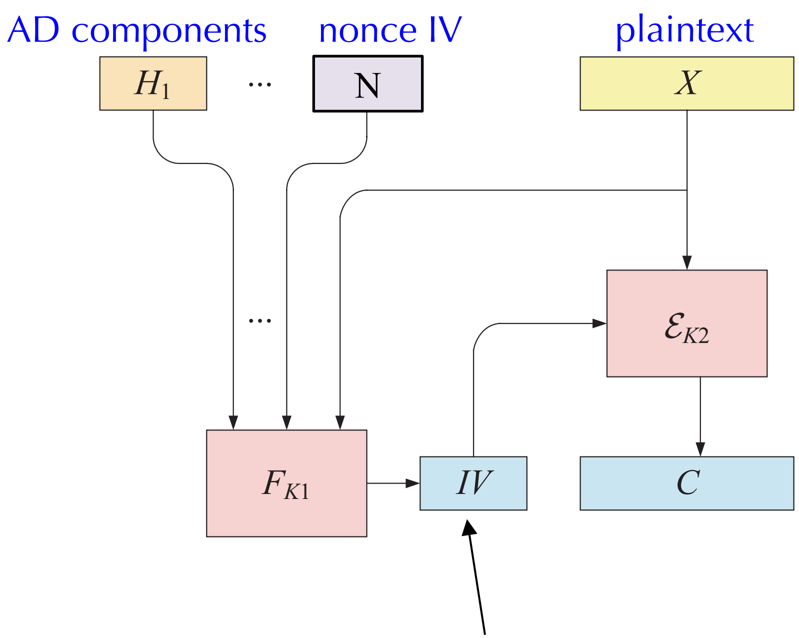
"E and M" "E and M" "E and M" SIV mode [RS06]



"E then M" "E then M" "M then E" "M then E"

Something cool about SIV: nonce-misuse resistance (MRAE)

(Rogaway, S. EC'06) (RFC 5297)

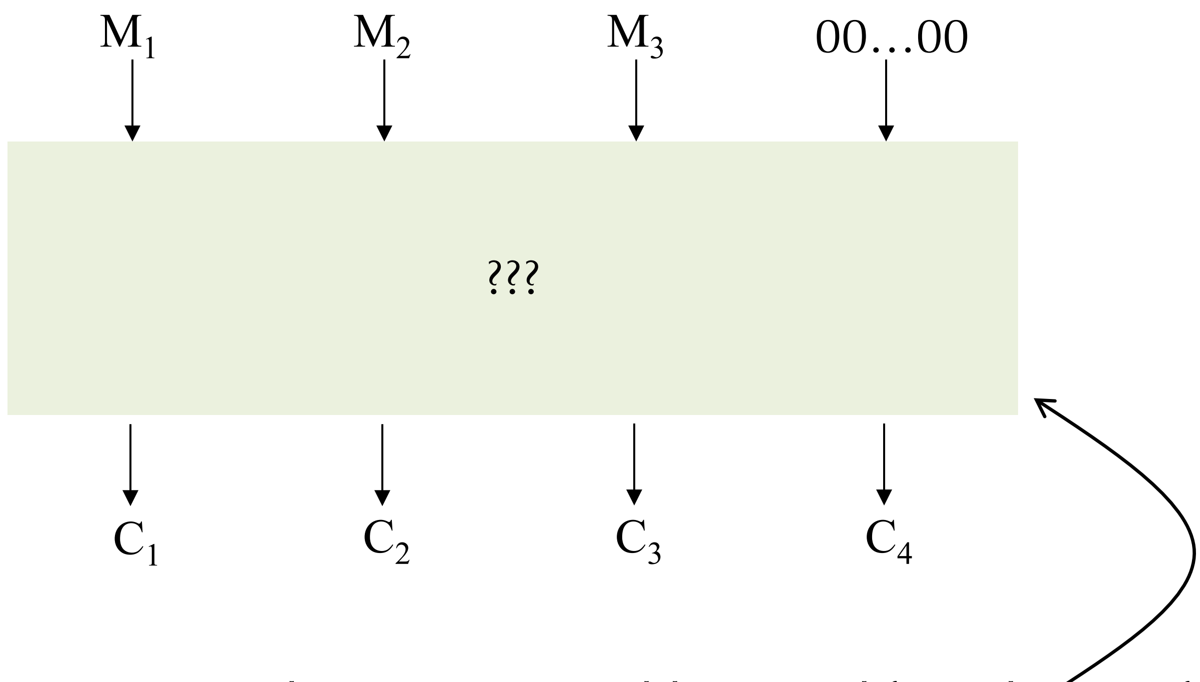


The synthetic IV depends on every bit of the input,
so as long as some bit of the input changes across encryptions,
we don't need to insist on non-repeating *N*

Many AE schemes (CCM, OCB, IAPM, CWC, GCM,…) fail catastrophically if the nonce/IV repeats, so this is a really nice feature.

Generic composition isn't the *only* path to AE

Let's revisit an idea from many slides ago…

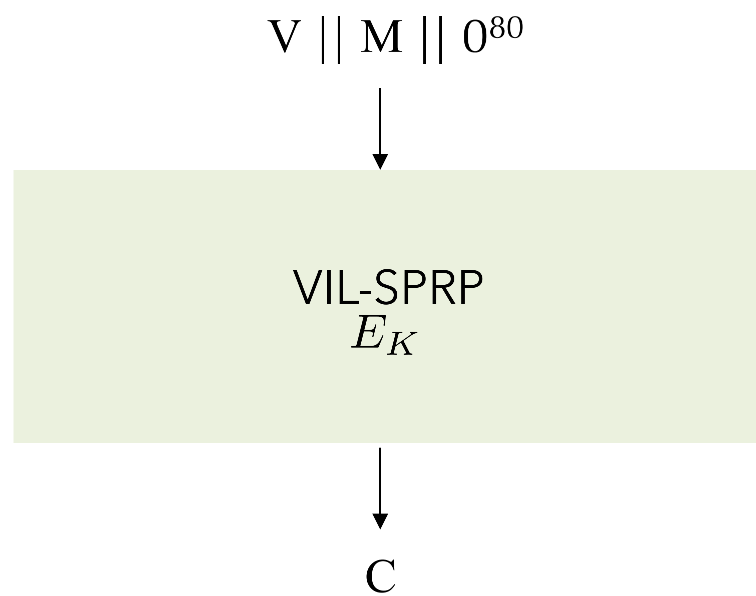


What property would we need from this transformation to get IND-CPA + INT-CTXT?

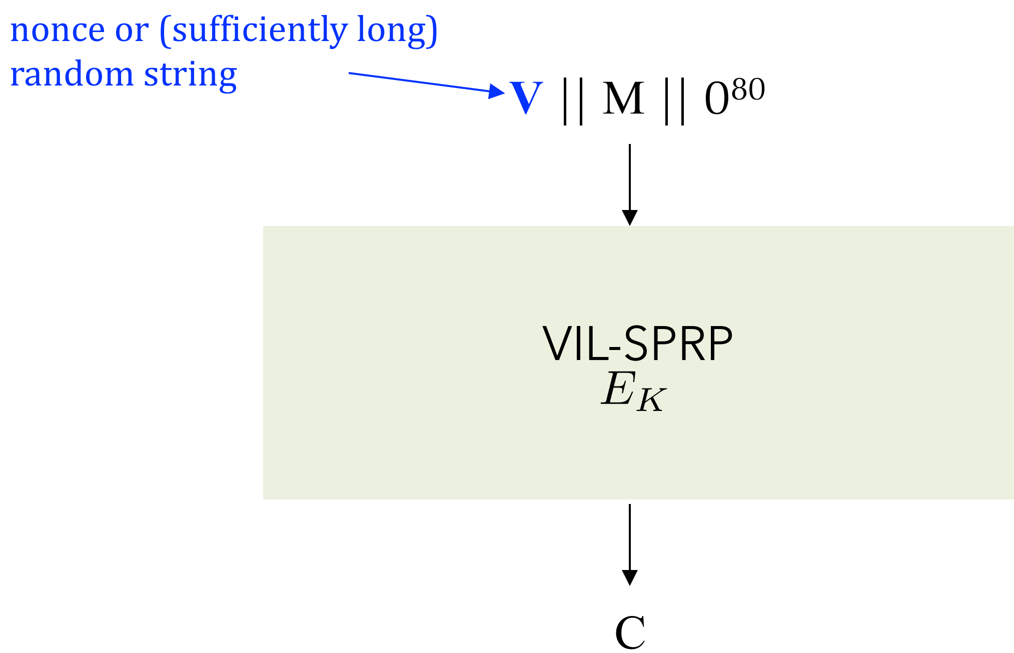
The "Encode-Encipher" paradigm

(Bellare, Rogaway AC' 00)

Length-preserving, "looks like" random permutation in both directions



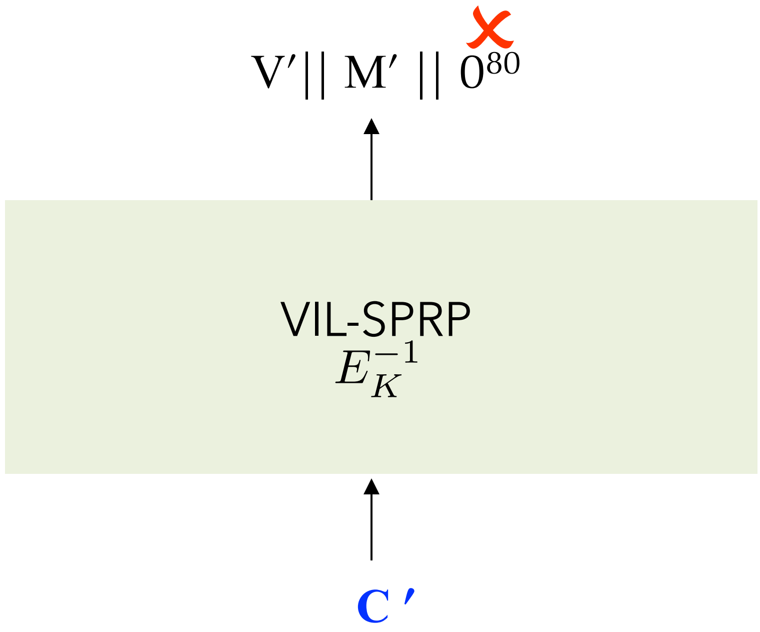
Privacy intuition: if you encrypt **distinct** inputs…



$(E_K \approx \pi)$

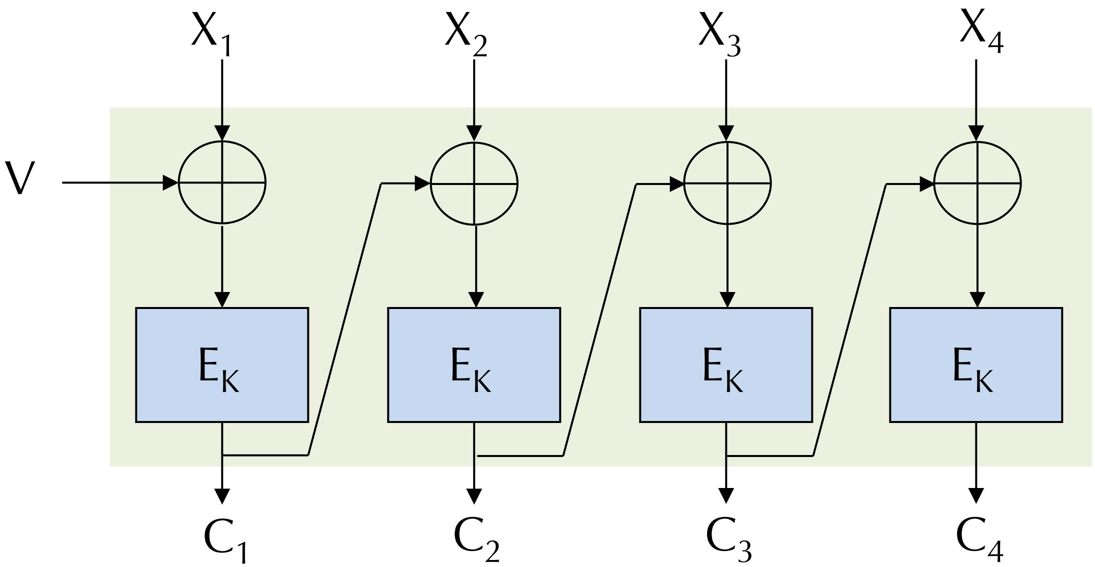
… then outputs look like **random bitstrings** (subject to permutivity)

Authenticity intuition: if you **flip any bit of a valid ciphertext** and decrypt…



$(E_K^{-1} \approx \pi^{-1})$

… then resulting “plaintext” **looks like a random bitstring**

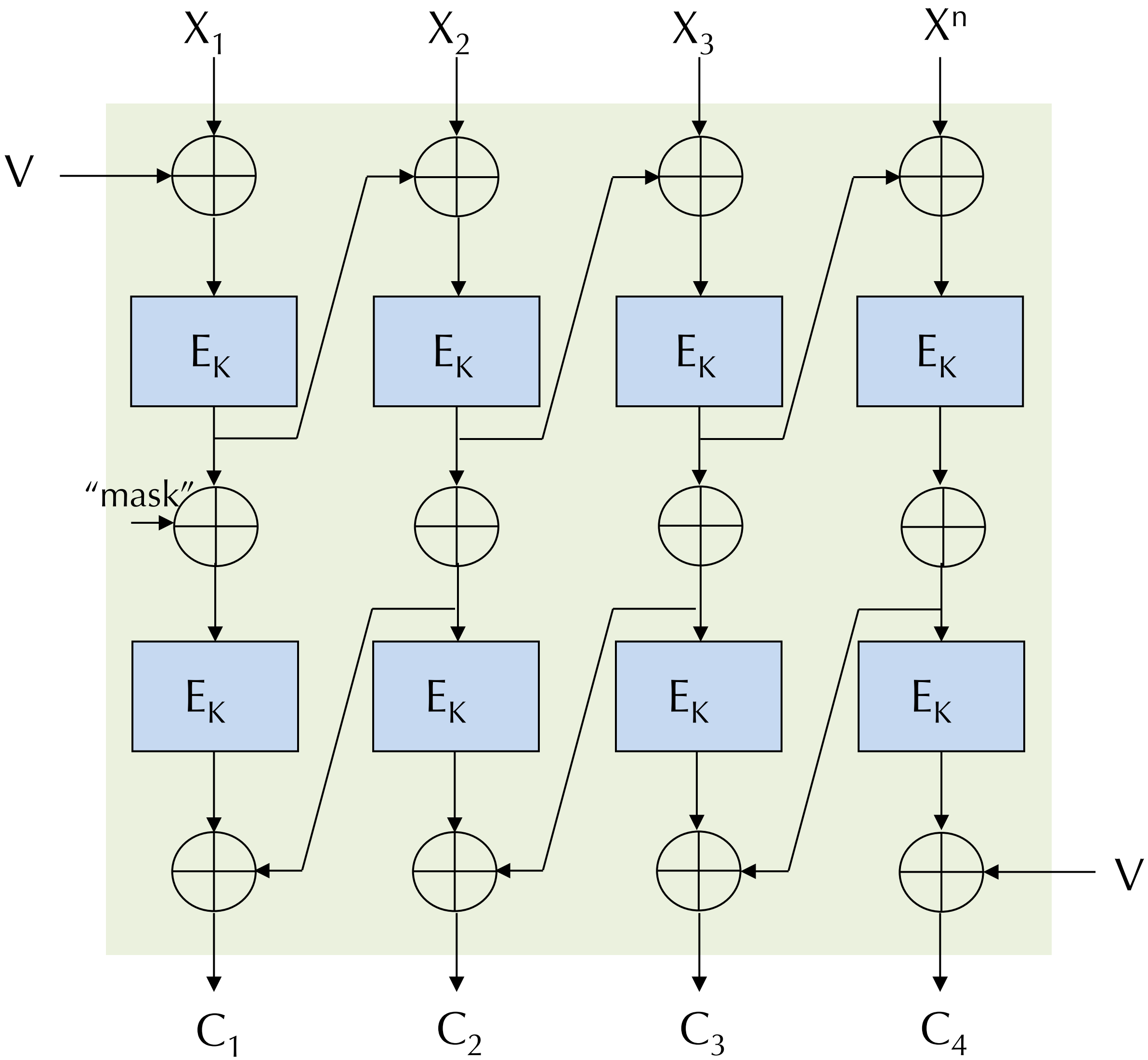


Definitely NOT an SPRP, even if E_K is.

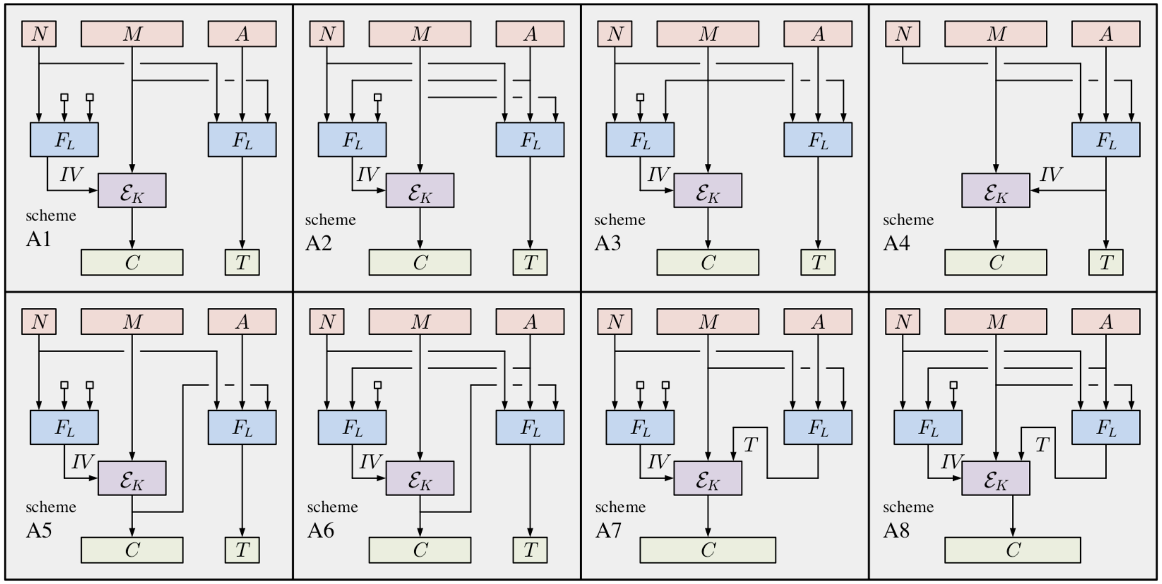
SPRPs require two full “cryptographic passes”

CMC mode

(Halevi, Rogaway C'03)



Two-pass vs. One-pass

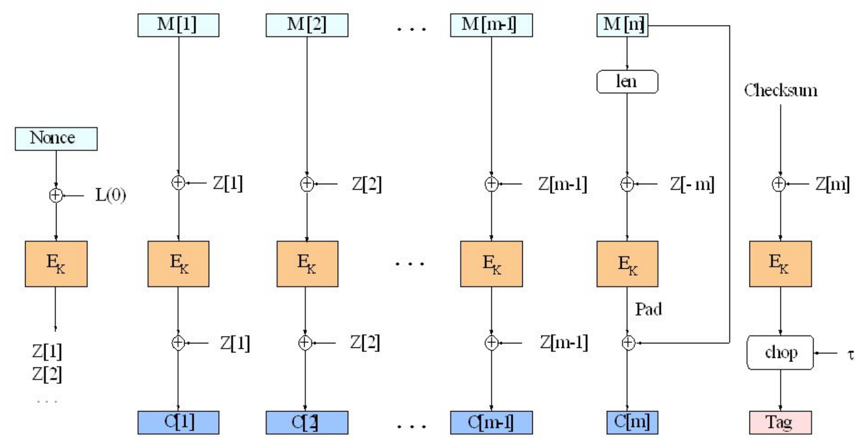


These schemes all require two "cryptographic passes"

(same for EAX, CCM, GCM-SIV, GCM (sort of), all SPRP-based schemes and many other "named" schemes…)

OCB mode

(Rogaway, Bellare, Black, Krovetz CCS'01)
(Rogaway AC'04)
(Krovetz, Rogaway FSE'11)
(Krovetz, Rogaway RFC 7253)

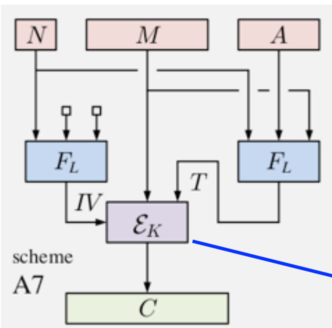


Checksum = M[1] ⊕M[2] ⊕ ··· ⊕M[m-1] ⊕C[m]0* ⊕Pad

Z[i] = Z[i-1] ⊕L(**ntz**(i))
L(0) = $E_K(\mathbf{0})$ and each L(i) obtained from L(i-1) by a shift and conditional xor

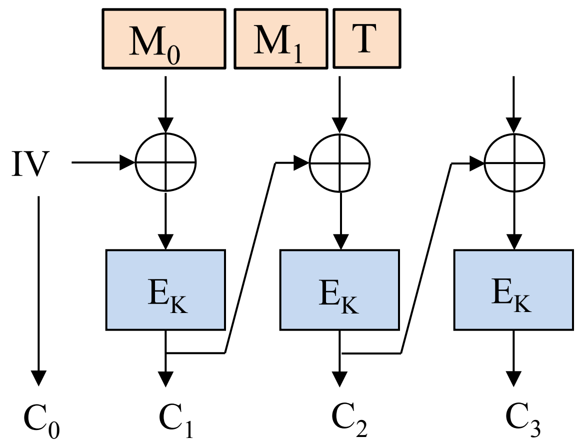
So… we've got this AEAD thing all locked up, right?

Provably AE secure



"M then E"

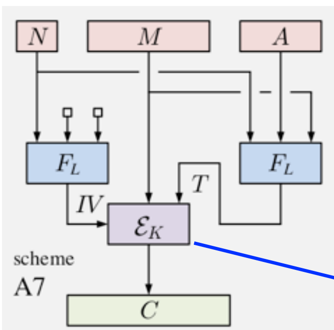
128 + 100 + 80 **= 308 bits**



What should we do?

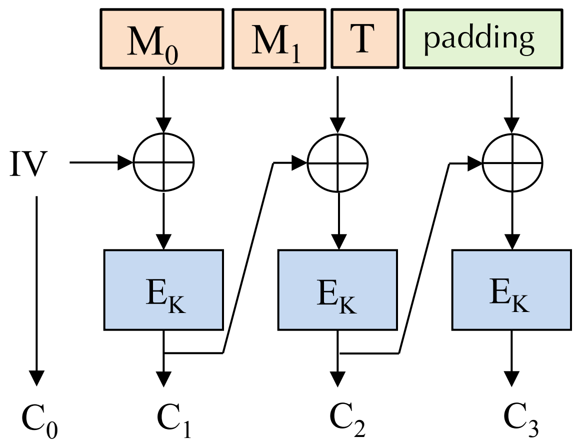
So… we've got this AEAD thing all locked up, right?

Provably AE secure



"M then E"

128 + 100 + 80 **= 308 bits**



What should we do?

Add padding!

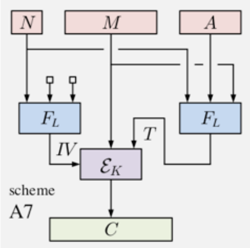
e.g. PKCS #7

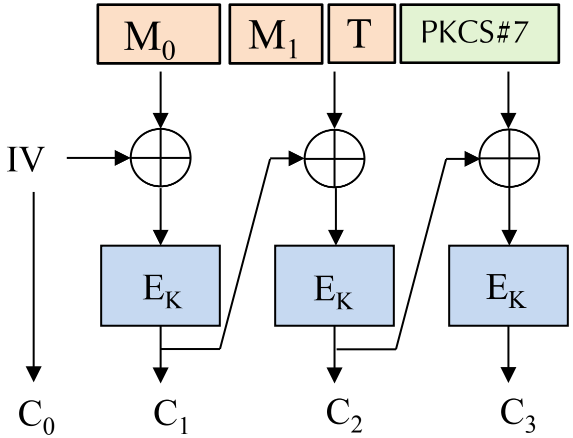
0x01

0x02 0x02

…

0xFF … 0xFF

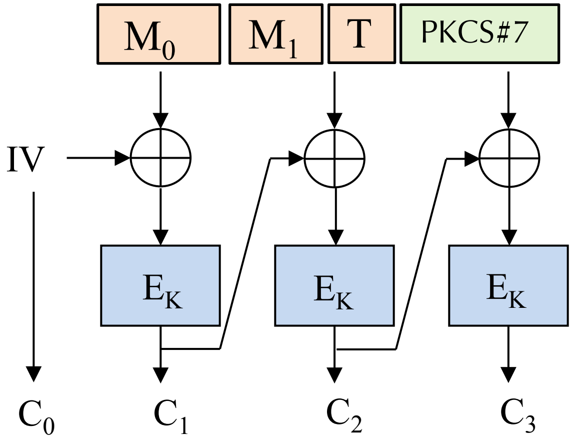


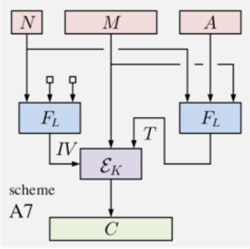


Decryption:

1. Should I check the padding? **If so, what should I do if it's incorrect?**
   Ignore it.
   Tell someone (who?) and continue.
   Tell someone (who?) and halt. ⟵ (what "halt" means depends a lot on the use-case, too…)
   Halt with no message.

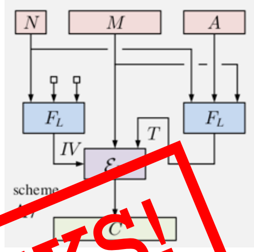
2. Should I check the tag T? (YES!) **What should I do if it's incorrect?**
   Ignore it.
   Tell someone (who?) and continue.
   Tell someone (who?) and halt.
   Halt with no message.

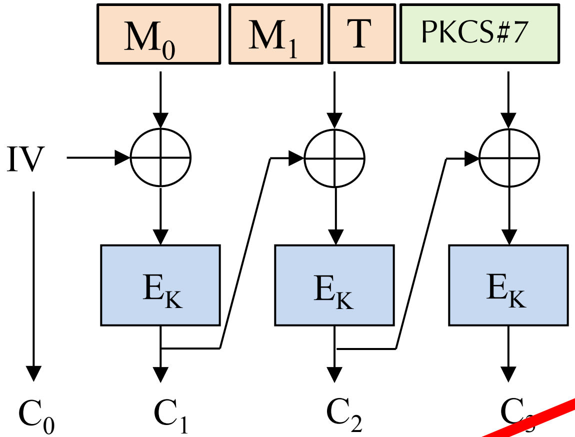




Decryption:

1. Check the padding.
   If invalid, surface a "**`bad padding`**" error and *continue processing* of this ciphertext.
2. Check the tag T.
   If invalid, surface a "**`bad tag`**" error and *halt processing of this ciphertext*.

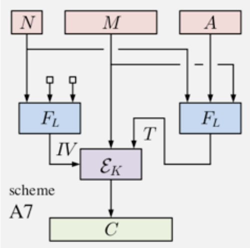




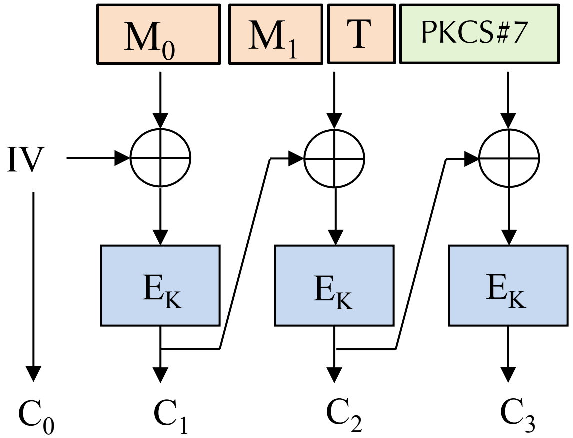
PADDING ORACLE ATTACKS!

Decryption:

1. Check the padding.
   If invalid, surface a "**`bad padding`**" error and *continue processing* of this ciphertext.
2. Check the tag T.
   If invalid, surface a "**`bad tag`**" error and *halt processing of this ciphertext*.



What happened?!



Decryption:

1. Check the padding.
   If invalid, surface a "**bad padding**" error and *continue processing* of this ciphertext.
2. Check the tag T.
   If invalid, surface a "**bad tag**" error and *halt processing of this ciphertext*.

Application of AEAD gives two distinguishable error messages, but syntax and security model only allow for one error message!

$$\mathcal{D}\colon (\mathcal{H} \times \mathcal{N}) \times \mathcal{K} \times \{0,1\}^* \to \{0,1\}^* \cup \{\bot\}$$

Padding oracle attacks are a **HUGE** problem in practice

...

SSL broken, again, in POODLE attack

Breaking Steam Client Cryptography

ASP.NET padding oracle vulnerability

Web Vulnerabilities / High Severity / ASP.NET padding oracle vulnerability

7 FEB 2013 NEWS

Lucky 13 – a new attack against SSL/TLS

« EXPLOITING F5 ICALL::SCRIPT PRIVILEGE ESCALATION (CVE-2015-3628) | MAIN | REVERSE SHELL OVER SMS (EXPLOITING CVE-2015-5897) »

Exploiting Padding Oracle To Gain Encryption Keys

Attack of the week: XML Encryption

CLOUDFLARE BLOG WHAT WE DO SUPPORT COMMUNITY

Yet Another Padding Oracle in OpenSSL CBC Ciphersuites

04 May 2016 by Filippo Valsorda.

IEEE P1735 Encryption Is Broken—Flaws Allow Intellectual Property Theft

Vendor Information (Learn More)

Vendor	Status	Date Notified	Date Updated
AMD	Unknown	03 Nov 2017	03 Nov 2017
Cadence Design Systems	Unknown	29 Sep 2017	29 Sep 2017
Cisco	Unknown	03 Nov 2017	03 Nov 2017
IBM, INC.	Unknown	03 Nov 2017	03 Nov 2017
Intel Corporation	Unknown	03 Nov 2017	03 Nov 2017
Marvell Semiconductors	Unknown	03 Nov 2017	03 Nov 2017
Mentor Graphics	Unknown	29 Sep 2017	29 Sep 2017
National Instruments (NI)	Unknown	03 Nov 2017	03 Nov 2017
National Semiconductor Corporation	Unknown	03 Nov 2017	03 Nov 2017
NXP Semiconductors Inc.	Unknown	03 Nov 2017	03 Nov 2017
QUALCOMM Incorporated	Unknown	03 Nov 2017	03 Nov 2017
Samsung Semiconductor Inc.	Unknown	03 Nov 2017	03 Nov 2017
Synopsys	Unknown	29 Sep 2017	29 Sep 2017
Xilinx	Unknown	29 Sep 2017	29 Sep 2017
Zuken Inc.	Unknown	29 Sep 2017	29 Sep 2017

Going beyond the basics

Security notions with multiple error messages	(Boldyreva, Degabriele, Paterson, Stam FSE'13)
"Release of unverified plaintext" (RUP)	(Andreeva, Bogdanov, Luykz, Mennink, Mouha, Yasuda AC'14)
AEAD in the presence of arbitrary "harmless" leakage (RUP plus a lot more)	(Hoang, Krovetz, Rogaway EC'15)
AEAD security in the presense of protocol leakage/side-channels	(Barwell, Martin, Oswald, Stam AC'17)
AEAD with ciphertext fragmentation	(Boldyreva, Degabriele, Paterson, Stam EC'12)
Online AE + nonce-misuse resistance	(Hoang, Reyhanitabar, Rogaway, Vizar EC'15)

(And don't forget constructions of AEAD from other primitives, e.g., wide permutations/sponges)

(A brief, incomplete)

Introduction to Authenticated Encryption

Tom Shrimpton

Summer School on Real-World Crypto and Privacy

June 11, 2018