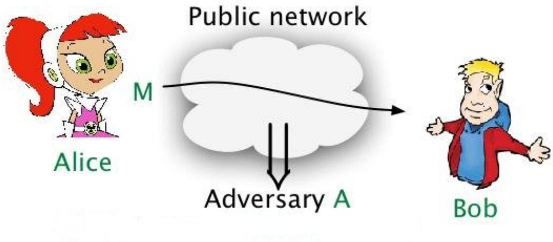
AUTHENTICATED ENCRYPTION

So Far ...



We have looked at methods to provide privacy and authenticity separately:

Goal	Primitive	Security notion
Data privacy	symmetric encryption	IND-CPA
Data authenticity	MAC	UF-CMA

Authenticated Encryption

In practice we often want both privacy and authenticity.

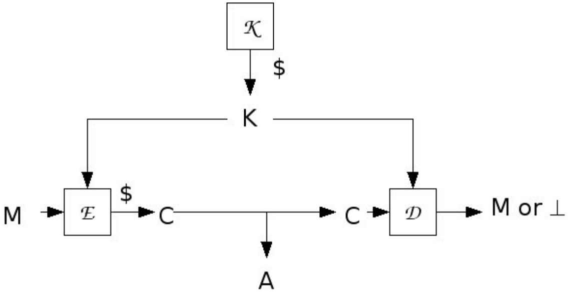
Example: A doctor wishes to send medical information M about Alice to the medical database. Then

- We want data privacy to ensure Alice's medical records remain confidential.
- We want authenticity to ensure the person sending the information is really the doctor and the information was not modified in transit.

We refer to this as authenticated encryption.

Authenticated Encryption Schemes

Syntactically, an authenticated encryption scheme is just a symmetric encryption scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ where



Privacy of Authenticated Encryption Schemes

The notion of privacy for symmetric encryption carries over, namely we want IND-CPA.

Integrity of Authenticated Encryption Schemes

Adversary's goal is to get the receiver to accept a "non-authentic" ciphertext C.

Integrity of ciphertexts: C is "non-authentic" if it was never transmitted by the sender.

INT-CTXT

Let $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme and A an adversary.

Game $\text{INTCTXT}_{\mathcal{AE}}$

procedure Initialize
$K \xleftarrow{\$} \mathcal{K}$; $S \leftarrow \emptyset$

procedure Enc(M)
$C \xleftarrow{\$} \mathcal{E}_K(M)$
$S \leftarrow S \cup \{C\}$
Return C

procedure Finalize(C)
$M \leftarrow \mathcal{D}_K(C)$
if $(C \notin S \wedge M \neq \perp)$ then
    return true
Else return false

The int-ctxt advantage of A is

$$\mathbf{Adv}^{\text{int-ctxt}}_{\mathcal{AE}}(A) = \Pr[\text{INTCTXT}^{A}_{\mathcal{AE}} \Rightarrow \text{true}]$$

Integrity with privacy

The goal of authenticated encryption is to provide both integrity and privacy. We will be interested in IND-CPA + INT-CTXT.

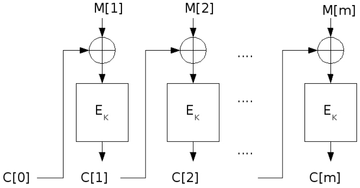
Plain Encryption Does Not Provide Integrity

Alg $\mathcal{E}_K(M)$

$C[0] \xleftarrow{\$} \{0,1\}^n$
For $i = 1, \ldots, m$ do
  $C[i] \leftarrow \mathrm{E}_K(C[i-1] \oplus M[i])$
Return C

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

For $i = 1, \ldots, m$ do
  $M[i] \leftarrow \mathrm{E}_K^{-1}(C[i]) \oplus C[i-1]$
Return M



Question: Is CBC\$ encryption INT-CTXT secure?

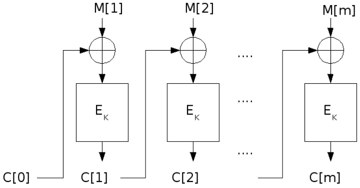
Plain Encryption Does Not Provide Integrity

Alg $\mathcal{E}_K(M)$

$C[0] \xleftarrow{\$} \{0,1\}^n$
For $i = 1, \ldots, m$ do
  $C[i] \leftarrow \mathrm{E}_K(C[i-1] \oplus M[i])$
Return C

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

For $i = 1, \ldots, m$ do
  $M[i] \leftarrow \mathrm{E}_K^{-1}(C[i]) \oplus C[i-1]$
Return M



Question: Is $\mathrm{CBC\$}$ encryption INT-CTXT secure?

Answer: No, because any string $C[0]C[1]\ldots C[m]$ has a valid decryption.

Plain Encryption Does Not Provide Integrity

Alg $\mathcal{E}_K(M)$

$C[0] \xleftarrow{\$} \{0,1\}^n$
For $i = 1, \ldots, m$ do
  $C[i] \leftarrow \mathsf{E}_K(C[i-1] \oplus M[i])$
Return C

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

For $i = 1, \ldots, m$ do
  $M[i] \leftarrow \mathsf{E}_K^{-1}(C[i]) \oplus C[i-1]$
Return M

adversary A

$C[0]C[1]C[2] \xleftarrow{\$} \{0,1\}^{3n}$
Return $C[0]C[1]C[2]$

Then

$$\mathbf{Adv}^{\text{int-ctxt}}_{\mathcal{SE}}(A) = 1$$

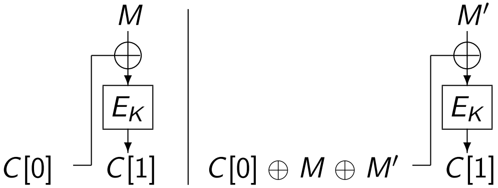
This violates INT-CTXT.

A scheme whose decryption algorithm never outputs $\perp$ cannot provide integrity!

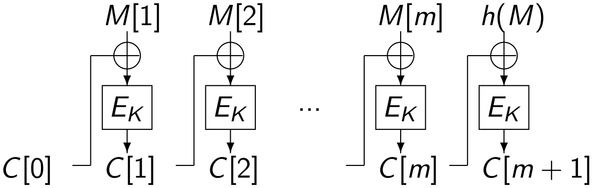
A Better Attack on CBC$

Suppose A has the CBC$ encryption $C[0]C[1]$ of a 1-block known message M. Then it can create an encryption $C'[0]C'[1]$ of *any* (1-block) message M' of its choice via

$C'[0] \leftarrow C[0] \oplus M \oplus M'$
$C'[1] \leftarrow C[1]$



Encryption with Redundancy

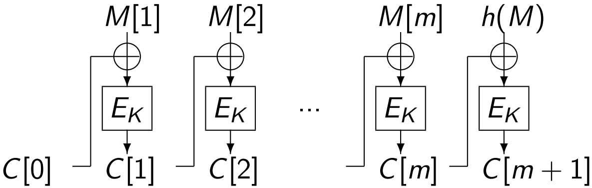


Here $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ is our block cipher and $h\colon \{0,1\}^* \to \{0,1\}^n$ is a "redundancy" function, for example

- $h(M[1] \ldots M[m]) = 0^n$
- $h(M[1] \ldots M[m]) = M[1] \oplus \cdots \oplus M[m]$
- A CRC
- $h(M[1] \ldots M[m])$ is the first n bits of SHA1$(M[1] \ldots M[m])$.

The redundancy is verified upon decryption.

Encryption with Redundancy



Let $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be our block cipher and $h\colon \{0,1\}^* \to \{0,1\}^n$ a redundancy function. Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}', \mathcal{D}')$ be CBC\$ encryption and define the encryption with redundancy scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ via

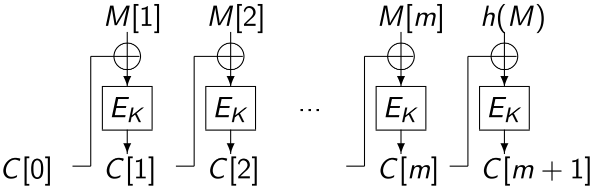
Alg $\mathcal{E}_K(M)$

$M[1]\ldots M[m] \leftarrow M$
$M[m+1] \leftarrow h(M)$
$C \stackrel{\$}{\leftarrow} \mathcal{E}'_K(M[1]\ldots M[m]M[m+1])$
return C

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

$M[1]\ldots M[m]M[m+1] \leftarrow \mathcal{D}'_K(C)$
if $(M[m+1] = h(M))$ then
    return $M[1]\ldots M[m]$
else return $\perp$

Arguments in Favor of Encryption with Redundancy

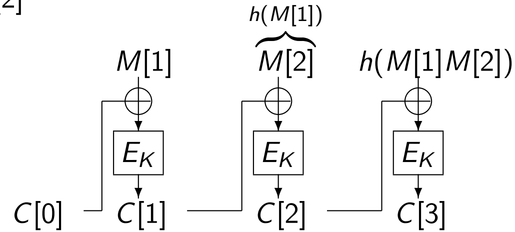


The adversary will have a hard time producing the last enciphered block of a new message.

Encryption with Redundancy Fails

adversary A

$M[1] \xleftarrow{\$} \{0,1\}^n$; $M[2] \leftarrow h(M[1])$
$C[0]C[1]C[2]C[3] \xleftarrow{\$} \mathbf{Enc}(M[1]M[2])$
Return $C[0]C[1]C[2]$



This attack succeeds for any (not secret-key dependent) redundancy function h.

WEP Attack

A "real-life" rendition of this attack broke the 802.11 WEP protocol, which instantiated h as CRC and used a stream cipher for encryption [BGW].

What makes the attack easy to see is having a clear, strong and formal security model.

Generic Composition

Build an authenticated encryption scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ by combining

- a given IND-CPA symmetric encryption scheme $\mathcal{SE} = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$
- a given PRF $F : \{0,1\}^k \times \{0,1\}^* \to \{0,1\}^n$

	CBC\$-AES	CTR\$-AES	...
HMAC-SHA1			
CMAC			
ECBC			
⋮			

Generic Composition

Build an authenticated encryption scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ by combining

- a given IND-CPA symmetric encryption scheme $\mathcal{SE} = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$
- a given PRF $F : \{0,1\}^k \times \{0,1\}^* \to \{0,1\}^n$

A key $K = K_e || K_m$ for $\mathcal{AE}$ always consists of a key K_e for $\mathcal{SE}$ and a key K_m for F:

Alg $\mathcal{K}$

$K_e \xleftarrow{\$} \mathcal{K}'$; $K_m \xleftarrow{\$} \{0,1\}^k$
Return $K_e || K_m$

Generic Composition Methods

The order in which the primitives are applied is important. Can consider

Method	Usage
Encrypt-and-MAC (E&M)	SSH
MAC-then-encrypt (MtE)	SSL/TLS
Encrypt-then-MAC (EtM)	IPSec

We study these following [BN].

Encrypt-and-MAC

$\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined by

$\underline{\textbf{Alg } \mathcal{E}_{K_e||K_m}(M)}$
$C' \xleftarrow{\$} \mathcal{E}'_{K_e}(M)$
$T \leftarrow F_{K_m}(M)$
Return $C'||T$

$\underline{\textbf{Alg } \mathcal{D}_{K_e||K_m}(C'||T)}$
$M \leftarrow \mathcal{D}'_{K_e}(C')$
If $(T = F_{K_m}(M))$ then return M
Else return $\perp$

Security	Achieved?
IND-CPA	
INT-CTXT	

Encrypt-and-MAC

$\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined by

Alg $\mathcal{E}_{K_e||K_m}(M)$

$C' \xleftarrow{\$} \mathcal{E}'_{K_e}(M)$
$T \leftarrow F_{K_m}(M)$
Return $C'||T$

Alg $\mathcal{D}_{K_e||K_m}(C'||T)$

$M \leftarrow \mathcal{D}'_{K_e}(C')$
If $(T = F_{K_m}(M))$ then return M
Else return $\perp$

Security	Achieved?
IND-CPA	NO
INT-CTXT	

Why? $T = F_{K_m}(M)$ is a deterministic function of M and allows detection of repeats.

Encrypt-and-MAC

$\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined by

$\underline{\textbf{Alg } \mathcal{E}_{K_e||K_m}(M)}$
$C' \xleftarrow{\$} \mathcal{E}'_{K_e}(M)$
$T \leftarrow F_{K_m}(M)$
Return $C'||T$

$\underline{\textbf{Alg } \mathcal{D}_{K_e||K_m}(C'||T)}$
$M \leftarrow \mathcal{D}'_{K_e}(C')$
If $(T = F_{K_m}(M))$ then return M
Else return $\perp$

Security	Achieved?
IND-CPA	NO
INT-CTXT	

Encrypt-and-MAC

$\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined by

Alg $\mathcal{E}_{K_e||K_m}(M)$

$C' \xleftarrow{\$} \mathcal{E}'_{K_e}(M)$
$T \leftarrow F_{K_m}(M)$
Return $C'||T$

Alg $\mathcal{D}_{K_e||K_m}(C'||T)$

$M \leftarrow \mathcal{D}'_{K_e}(C')$
If $(T = F_{K_m}(M))$ then return M
Else return $\perp$

Security	Achieved?
IND-CPA	NO
INT-CTXT	NO

Why? May be able to modify C' in such a way that its decryption is unchanged.

MAC-then-Encrypt

$\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined by

Alg $\mathcal{E}_{K_e||K_m}(M)$

$T \leftarrow F_{K_m}(M)$
$C \xleftarrow{\$} \mathcal{E}'_{K_e}(M||T)$
Return C

Alg $\mathcal{D}_{K_e||K_m}(C)$

$M||T \leftarrow \mathcal{D}'_{K_e}(C)$
If $(T = F_{K_m}(M))$ then return M
Else return $\perp$

Security	Achieved?
IND-CPA	
INT-CTXT	

MAC-then-Encrypt

$\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined by

Alg $\mathcal{E}_{K_e||K_m}(M)$

$T \leftarrow F_{K_m}(M)$
$C \xleftarrow{\$} \mathcal{E}'_{K_e}(M||T)$
Return C

Alg $\mathcal{D}_{K_e||K_m}(C)$

$M||T \leftarrow \mathcal{D}'_{K_e}(C)$
If $(T = F_{K_m}(M))$ then return M
Else return $\perp$

Security	Achieved?
IND-CPA	YES
INT-CTXT	

Why? $\mathcal{SE}' = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$ is IND-CPA secure.

MAC-then-Encrypt

$\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined by

Alg $\mathcal{E}_{K_e||K_m}(M)$

$T \leftarrow F_{K_m}(M)$
$C \stackrel{\$}{\leftarrow} \mathcal{E}'_{K_e}(M||T)$
Return C

Alg $\mathcal{D}_{K_e||K_m}(C)$

$M||T \leftarrow \mathcal{D}'_{K_e}(C)$
If $(T = F_{K_m}(M))$ then return M
Else return $\perp$

Security	Achieved?
IND-CPA	YES
INT-CTXT	

MAC-then-Encrypt

$\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined by

Alg $\mathcal{E}_{K_e||K_m}(M)$

$T \leftarrow F_{K_m}(M)$
$C \xleftarrow{\$} \mathcal{E}'_{K_e}(M||T)$
Return C

Alg $\mathcal{D}_{K_e||K_m}(C)$

$M||T \leftarrow \mathcal{D}'_{K_e}(C)$
If $(T = F_{K_m}(M))$ then return M
Else return $\perp$

Security	Achieved?
IND-CPA	YES
INT-CTXT	NO

Why? May be able to modify C in such a way that its decryption is unchanged.

Encrypt-then-MAC

$\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined by

Alg $\mathcal{E}_{K_e \| K_m}(M)$	**Alg** $\mathcal{D}_{K_e \| K_m}(C' \| T)$
$C' \xleftarrow{\$} \mathcal{E}'_{K_e}(M)$	$M \leftarrow \mathcal{D}'_{K_e}(C')$
$T \leftarrow F_{K_m}(C')$	If $(T = F_{K_m}(C'))$ then return M
Return $C' \| T$	Else return $\perp$

Security	Achieved?
IND-CPA	
INT-CTXT	

Encrypt-then-MAC

$\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined by

Alg $\mathcal{E}_{K_e \| K_m}(M)$	**Alg** $\mathcal{D}_{K_e \| K_m}(C' \| T)$
$C' \xleftarrow{\$} \mathcal{E}'_{K_e}(M)$	$M \leftarrow \mathcal{D}'_{K_e}(C')$
$T \leftarrow F_{K_m}(C')$	If $(T = F_{K_m}(C'))$ then return M
Return $C' \| T$	Else return $\perp$

Security	Achieved?
IND-CPA	YES
INT-CTXT	

Why? $\mathcal{SE}' = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$ is IND-CPA secure.

Encrypt-then-MAC

$\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined by

Alg $\mathcal{E}_{K_e \| K_m}(M)$	**Alg** $\mathcal{D}_{K_e \| K_m}(C' \| T)$
$C' \xleftarrow{\$} \mathcal{E}'_{K_e}(M)$	$M \leftarrow \mathcal{D}'_{K_e}(C')$
$T \leftarrow F_{K_m}(C')$	If $(T = F_{K_m}(C'))$ then return M
Return $C' \| T$	Else return $\perp$

Security	Achieved?
IND-CPA	YES
INT-CTXT	

Encrypt-then-MAC

$\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined by

Alg $\mathcal{E}_{K_e||K_m}(M)$

$C' \xleftarrow{\$} \mathcal{E}'_{K_e}(M)$
$T \leftarrow F_{K_m}(C')$
Return $C'||T$

Alg $\mathcal{D}_{K_e||K_m}(C'||T)$

$M \leftarrow \mathcal{D}'_{K_e}(C')$
If $(T = F_{K_m}(C'))$ then return M
Else return $\perp$

Security	Achieved?
IND-CPA	YES
INT-CTXT	YES

Why? If $C||T$ is new then T will be wrong.

Two keys or one?

We have used separate keys K_e, K_m for the encryption and message authentication. However, these can be derived from a single key K via $K_e = F_K(0)$ and $K_m = F_K(1)$, where F is a PRF such as a block cipher, the CBC-MAC or HMAC.

Trying to directly use the same key for the encryption and message authentication is error-prone, but works if done correctly.

Exercise: Setup

Alg $\underline{\mathcal{E}_K(M)}$

if $|M| \neq 512$ then return $\perp$

$M[1] \ldots M[4] \leftarrow M$

$C_e[0] \xleftarrow{\$} \{0,1\}^{128} C_m[0] \leftarrow 0^{128}$

for $i = 1, \ldots, 4$ do

$\quad C_e[i] \leftarrow E_K(C_e[i-1] \oplus M[i])$

$\quad C_m[i] \leftarrow E_K(C_m[i-1] \oplus M[i])$

$C_e \leftarrow C_e[0]C_e[1]C_e[2]C_e[3]C_e[4]$

$T \leftarrow C_m[4]$; return (C_e, T)

Alg $\underline{\mathcal{D}_K((C_e, T))}$

if $|C_e| \neq 640$ then return $\perp$

$C_m[0] \leftarrow 0^{128}$

for $i = 1, \ldots, 4$ do

$\quad M[i] \leftarrow E_K^{-1}(C_e[i]) \oplus C_e[i-1]$

$\quad C_m[i] \leftarrow E_K(C_m[i-1] \oplus M[i])$

if $C_m[4] \neq T$ then return $\perp$

return M

Let $E = AES$. Let $\mathcal{K}$ return a random 128-bit AES key K. Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ where $\mathcal{E}$, $\mathcal{D}$ are above. Here, $X[i]$ denotes the i-th 128-bit block of a string whose length is a multiple of 128.

Exercise: Questions

1. Is $\mathcal{SE}$ IND-CPA-secure? Why or why not?
2. Is $\mathcal{SE}$ INT-CTXT-secure? Why or why not?
3. Is $\mathcal{SE}$ an Encrypt-and-MAC construction? Justify your answer.

Exercise

You are given

- An IND-CPA symmetric encryption scheme $\mathcal{SE}^* = (\mathcal{K}^*, \mathcal{E}^*, \mathcal{D}^*)$
- A PRF F: $\{0,1\}^k \times \{0,1\}^* \rightarrow \{0,1\}^n$

Construct a symmetric encryption scheme $\mathcal{SE}' = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$ such that

(1) $\mathcal{SE}'$ is IND-CPA, but

(2) The MtE combination of $\mathcal{SE}'$ and F is not INT-CTXT-secure.

Specify $\mathcal{SE}'$ by giving pseudocode for all the constituent algorithms.

Then prove (1) by a reduction and prove (2) by giving pseudocode for an efficient adversary achieving int-ctxt advantage 1.

INT-CTXT security of Encrypt-then-MAC

Encrpt-then-MAC is INT-CTXT-secure assuming PRF-security of F:

Theorem: Let $\mathcal{SE} = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$ be a symmetric encryption scheme. Let $F : \{0,1\}^k \times \{0,1\}^* \to \{0,1\}^n$ be a family of functions. Let $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be obtained by composing $\mathcal{SE}$ and F in the Encrypt-then-MAC combination. Let A be an int-ctxt adversary against $\mathcal{AE}$ make q_e **Enc** queries and having running time t. Then we can construct a prf-adversary B against F such that

$$\mathbf{Adv}^{\mathrm{int\text{-}ctxt}}_{\mathcal{AE}}(A) \leq \mathbf{Adv}^{\mathrm{prf}}_{F}(B) + \frac{1}{2^n} .$$

B makes q_e queries to its **Fn** oracle and has running time t plus some overhead.

The adversary B

adversary B

$K_e \xleftarrow{\$} \mathcal{K}'$; $S \leftarrow \emptyset$
$C' \| T \xleftarrow{\$} A^{\mathrm{EncSim}}$
If $(C', T) \in S$ then return 0
If $T = \mathbf{Fn}(C')$ then return 1
Else return 0

Subroutine $\mathrm{EncSim}(M)$
$C' \xleftarrow{\$} \mathcal{E}'(K_e, M)$; $T \leftarrow \mathbf{Fn}(C')$
$S \leftarrow S \cup \{(C', T')\}$
Return $C' \| T$

Note that B itself picks K_e so that it can simulate **Enc** for A.

$$\begin{aligned} \Pr[\mathrm{Real}_F^B \Rightarrow 1] &= \mathbf{Adv}^{\mathrm{int\text{-}ctxt}}_{\mathcal{AE}}(A) \\ \Pr[\mathrm{Rand}_{\{0,1\}^n}^B \Rightarrow 1] &\leq \frac{1}{2^n} \end{aligned}$$

Exercise

There is a lot going on in the above proof! The exercise is to work through it slowly, checking each step and claim.

Exercise: IND-CPA security of Encrypt-then-MAC

Encrpt-then-MAC is IND-CPA-secure assuming IND-CPA-security of $\mathcal{SE}'$:

Theorem: Let $\mathcal{SE} = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$ be a symmetric encryption scheme. Let $F : \{0,1\}^k \times \{0,1\}^* \to \{0,1\}^n$ be a family of functions. Let $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be obtained by composing $\mathcal{SE}$ and F in the Encrypt-then-MAC combination. Let A be an ind-cpa adversary against $\mathcal{AE}$ make q **LR** queries and having running time t. Then we can construct an ind-cpa adversary B against $\mathcal{SE}'$ such that

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}}(A) \leq \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}'}(B) \, .$$

B makes q queries to its **LR** oracle and has running time t plus some overhead.

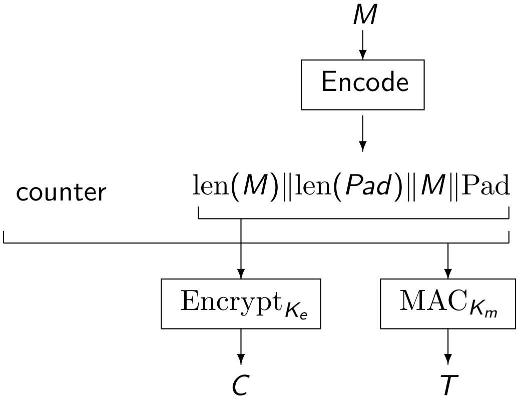
The exercise is to prove this theorem.

Generic Composition in Practice

AE in	**is based on**	**which in general is**	**and in this case is**
SSH	E&M	insecure	secure
SSL	MtE	insecure	insecure
SSL + RFC 4344	MtE	insecure	secure
IPSec	EtM	secure	secure
WinZip	EtM	secure	insecure

Why?

- Encodings
- Specific "E" and "M" schemes
- For WinZip, disparity between usage and security model



SSH2 encryption uses inter-packet chaining which is insecure [D, BKN]. RFC 4344 [BKN] proposed fixes that render SSH provably IND-CPA + INT-CTXT secure. Fixes recommended by Secure Shell Working Group and included in OpenSSH since 2003. Fixes included in PuTTY since 2008.

AE in SSL

SSL uses MtE

$$\mathcal{E}_{K_e \| K_M} = \mathcal{E}'_{K_e}(M \| F_{K_m}(M))$$

which we saw is not INT-CTXT-secure in general. But $\mathcal{E}'$ is CBC$ in SSL, and in this case the scheme does achieve INT-CTXT [K].

F in SSL is HMAC.

Sometimes SSL uses RC4 for encryption.

AEAD

The goal has evolved into Authenticated Encryption with Associated Data (AEAD) [Ro].

- Associated Data (AD) is authenticated but not encrypted
- Schemes are nonce-based (and deterministic)

Sender

- $C \leftarrow \mathcal{E}_K(N, AD, M)$
- Send (N, AD, C)

Receiver

- Receive (N, AD, C)
- $M \leftarrow \mathcal{D}_K(N, AD, C)$

Sender must never re-use a nonce.

But when attacking integrity, the adversary may use any nonce it likes.

AEAD Schemes

Generic composition: E&M, MtE, EtM extend and again EtM is the best but others work too under appropriate conditions [NRS14].

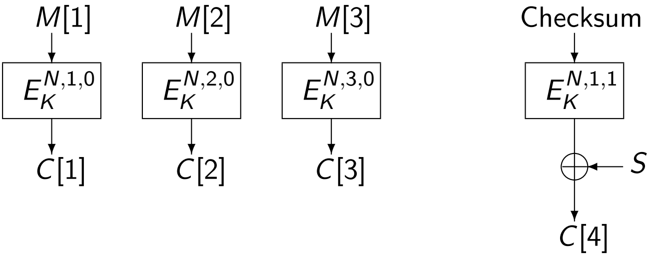
1-pass schemes: IAPM [J], XCBC/XEBC [GD], OCB [RBBK, R]

2-pass schemes: CCM [FHW], EAX [BRW], CWC [KVW], GCM [MV]

Stream cipher based: Helix [FWSKLK], SOBER-128 [HR]

- 1-pass schemes are fast
- 2-pass schemes are patent-free
- Stream cipher based schemes are fast

OCB [RBBK]



$\mathrm{Checksum} = M[1] \oplus M[2] \oplus M[3]$
$S = \mathrm{PMAC}_K(AD)$ using separate tweaks.
Output may optionally be truncated.
Some complications (not shown) for non-full messages.

Optional in IEEE 802.11i

Patents on 1-pass schemes

- Jutla (IBM) 7093126
- Gligor and Donescu (VDG, Inc.) 6973187
- Rogaway 7046802, 7200227

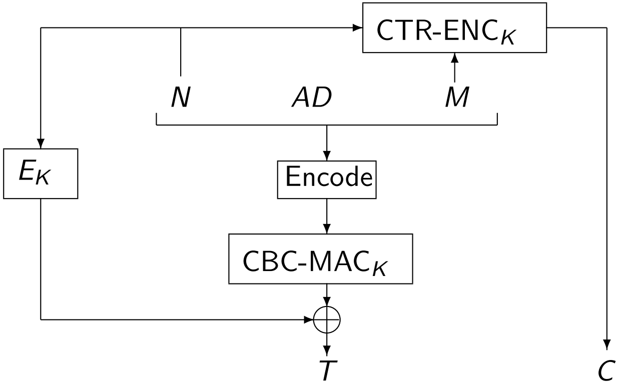
2-pass AEAD

- Tailored generic composition of specific base schemes
- Single key

Philosophical questions:

- What is the advantage of one key versus two given that can always derive the two from the one?
- Why not just do specific generic composition of specific base schemes?

CCM [FHW]

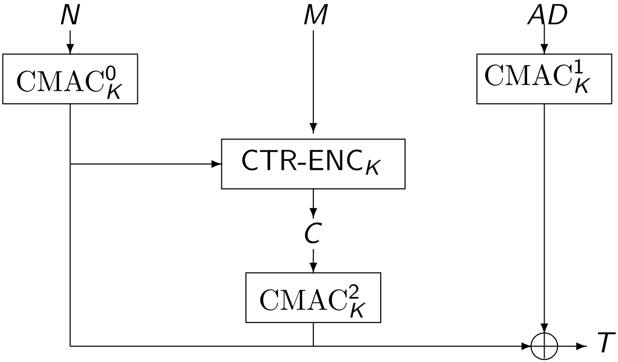


MtE-based but single key throughout. CTR-ENC is nonce-based counter mode encryption, and CBC-MAC is the basic CBC MAC. Ciphertext is $C \| T$. In NIST SP 800-38C, IEEE 802.11i.

Critiques of CCM [RW]

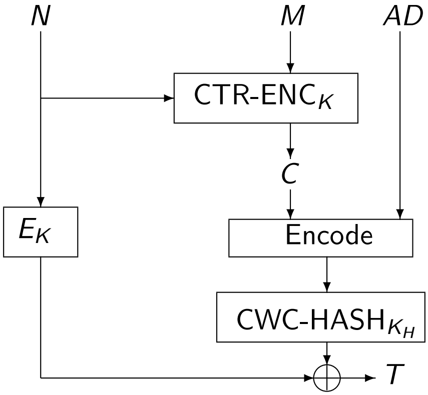
- Not on-line: message and AD lengths must be known in advance
- Can't pre-process static AD
- Nonce length depends on message length and the former decreases as the latter increases
- Awkward/unnecessary parameters
- Complex encodings

EAX [BRW]



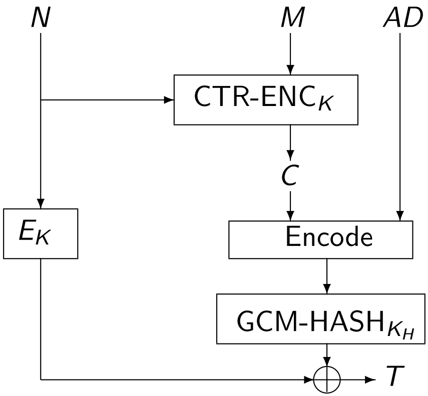
EtM-based but single key throughout. CTR-ENC is nonce-based counter mode encryption. Online; can pre-process static AD; always 128-bit nonce; simple; same performance as CCM. In ANSI C12.22.

CWC [KVW]



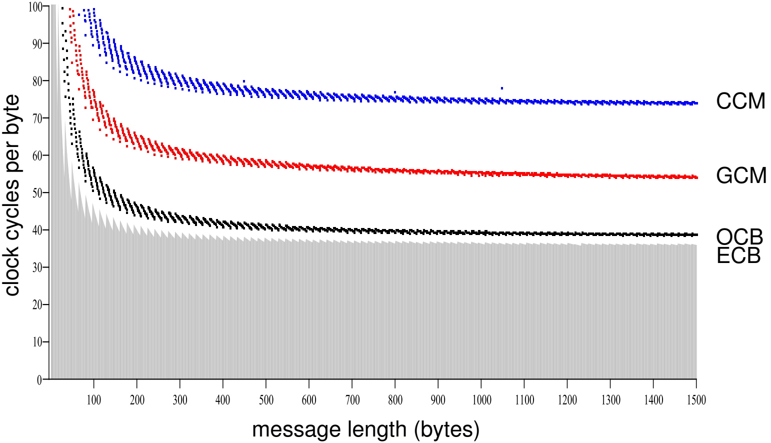
CTR-ENC is nonce-based counter mode encryption. CWC-HASH is a AU polynomial-based hash. K_H is derived from K via E. Parallelizable; 300K gates for 10 Gbit/s (ASIC at 130 nanometers); Roughly same software speed as CCM, EAX, but can be improved via precomputation.

GCM [MV]



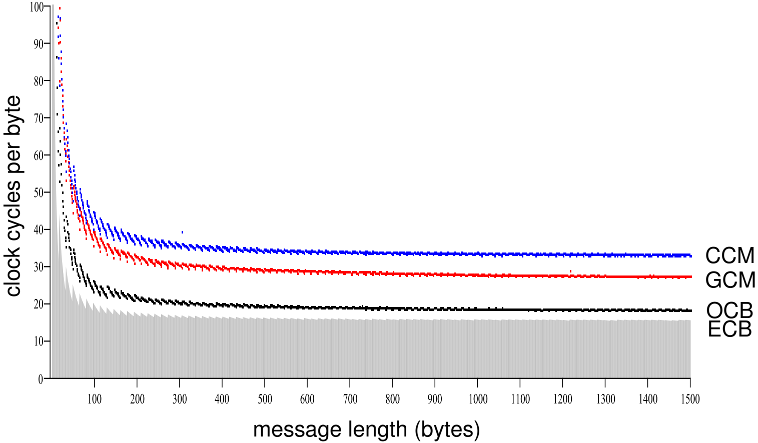
CTR-ENC is nonce-based counter mode encryption. GCM-HASH is a AU polynomial-based hash. K_H is derived from K via E. Can be used as a MAC. In NIST SP 800-38D.

Performance Comparisons x32



Gladman's C code

Performance Comparisons x64



Gladman's C code

Which AEAD scheme should I use?

No clear answer. Ask yourself

- What performance do I need?
- Single or multiple keys?
- Patents ok or not?
- Do I need to comply with some standard?

Authenticated encryption today

- The most important practical goal
- Lots of schemes, standards and implementations
- Big efforts go into making it FAST
- CAESAR competition:
  `http://competitions.cr.yp.to/caesar.html`