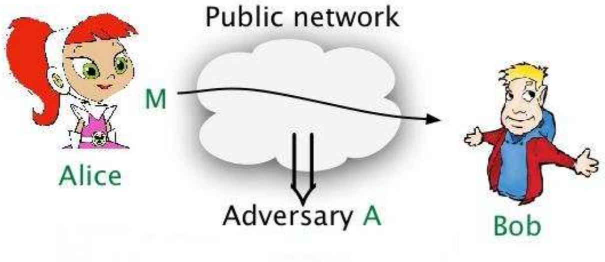
AUTHENTICATED ENCRYPTION

So Far ...



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

Goal	Primitive	Security notions
Data privacy	symmetric encryption	IND-CPA, IND-CCA
Data integrity/authenticity	MA scheme/MAC	UF-CMA, SUF-CMA

Authenticated Encryption

In practice we often want both privacy and integrity/authenticity.

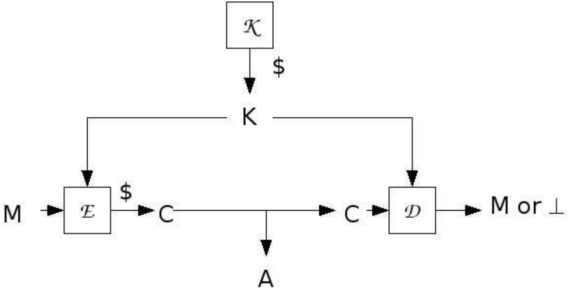
Example: A doctor wishes to send medical information M about Alice to the medical database. Then

- We want data privacy to ensure Alice's medical records remain confidential.
- We want integrity/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 notions of privacy for symmetric encryption carry over:

- IND-CPA
- IND-CCA

Integrity of Authenticated Encryption Schemes

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

Two possible interpretations of "non-authentic:"

- Integrity of plaintexts: $M = \mathcal{D}_K(C)$ was never encrypted by the sender
- Integrity of ciphertexts: C was never transmitted by the sender

INT-PTXT

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

Game $\text{INTPTXT}_{\mathcal{AE}}$	**procedure Dec**(C)
procedure Initialize	$M \leftarrow \mathcal{D}_K(C)$
$K \xleftarrow{\$} \mathcal{K}$; $S \leftarrow \emptyset$	if $(M \notin S \wedge M \neq \bot)$ then
procedure Enc(M)	$\quad \text{win} \leftarrow \text{true}$
$C \xleftarrow{\$} \mathcal{E}_K(M)$	return win
$S \leftarrow S \cup \{M\}$	**procedure Finalize**
return C	return win

The int-ptxt advantage of A is

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

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 Dec**(C)
procedure Initialize	$M \leftarrow \mathcal{D}_K(C)$
$K \stackrel{\$}{\leftarrow} \mathcal{K}$; $S \leftarrow \emptyset$	if $(C \notin S \wedge M \neq \perp)$ then
procedure Enc(M)	$\quad$ win $\leftarrow$ true
$C \stackrel{\$}{\leftarrow} \mathcal{E}_K(M)$	return win
$S \leftarrow S \cup \{C\}$	**procedure Finalize**
return C	return win

The int-ctxt advantage of A is

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

INT-CTXT $\Rightarrow$ INT-PTXT

If $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is INT-CTXT secure then it is also INT-PTXT secure.

Why? Suppose A makes **Enc** queries $M_1, \ldots, M_q$ resulting in ciphertexts

$$C_1 \xleftarrow{\$} \mathcal{E}_K(M_1), \ldots, C_q \xleftarrow{\$} \mathcal{E}_K(M_q)$$

suppose A makes query $\mathbf{Dec}(C)$, and let $M = \mathcal{D}_K(C)$.

Fact: $M \notin \{M_1, \ldots, M_q\} \Rightarrow C \notin \{C_1, \ldots, C_q\}$

So if A wins INT-PTXT$_{\mathcal{AE}}$ it also wins INT-CTXT$_{\mathcal{AE}}$.

Theorem: *For any adversary A,*

$$\mathbf{Adv}^{\text{int-ptxt}}_{\mathcal{AE}}(A) \leq \mathbf{Adv}^{\text{int-ctxt}}_{\mathcal{AE}}(A).$$

INT-PTXT $\not\Rightarrow$ INT-CTXT

Counterexample: Construct $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ which is

- not INT-CTXT secure, but
- is INT-PTXT secure

Approach: Start from some INT-PTXT secure $\mathcal{AE}' = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$ and modify it to $\mathcal{AE}$ so that:

- There is an attack showing $\mathcal{AE}$ is not INT-CTXT secure
- There is a proof by reduction showing $\mathcal{AE}$ inherits the INT-PTXT security of $\mathcal{AE}'$.

INT-PTXT $\not\Rightarrow$ INT-CTXT

Given $\mathcal{AE}' = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$, let $\mathcal{AE} = (\mathcal{K}', \mathcal{E}, \mathcal{D})$ where

Alg $\mathcal{E}_K(M)$	**Alg** $\mathcal{D}_K(C)$
$C' \xleftarrow{\$} \mathcal{E}'_K(M)$; $C \leftarrow 0 \Vert C'$	$b \Vert C' \leftarrow C$; $M \leftarrow \mathcal{D}'_K(C')$
Return C	Return M

Observe: If $C = 0 \Vert C' \xleftarrow{\$} \mathcal{E}_K(M)$ then

- $1 \Vert C' \neq 0 \Vert C'$, but
- $\mathcal{D}_K(1 \Vert C') = \mathcal{D}_K(0 \Vert C')$

adversary A

Let M be any message

$0 \Vert C' \xleftarrow{\$}$ **Enc**(M); $x \leftarrow$ **Dec**$(1 \Vert C')$

Then $\mathbf{Adv}^{\mathrm{int\text{-}ctxt}}_{\mathcal{AE}}(A) = 1$.

Note: This does not compromise INT-PTXT security because $x = M$.

INT-PTXT $\not\Rightarrow$ INT-CTXT

Given $\mathcal{AE}' = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$, let $\mathcal{AE} = (\mathcal{K}', \mathcal{E}, \mathcal{D})$ where

Alg $\mathcal{E}_K(M)$	**Alg** $\mathcal{D}_K(C)$
$C' \stackrel{\$}{\leftarrow} \mathcal{E}'_K(M)$; $C \leftarrow 0 \| C'$	$b \| C' \leftarrow C$; $M \leftarrow \mathcal{D}'_K(C')$
Return C	Return M

Claim: *If $\mathcal{AE}'$ is* INT-PTXT *secure, then so is $\mathcal{AE}$.*

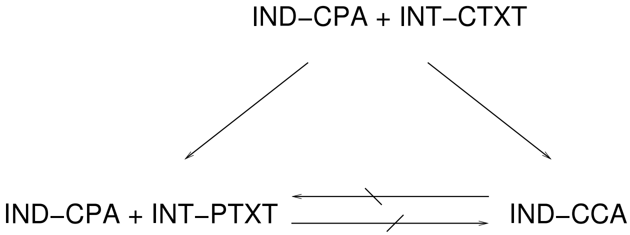
Why? An attack on $\mathcal{AE}$ can be turned into one on $\mathcal{AE}'$. A formal proof is by reduction.

Integrity with privacy

The goal of authenticated encryption is to provide both integrity and privacy. We will be interested in:

- IND-CPA + INT-PTXT
- IND-CPA + INT-CTXT

Relations

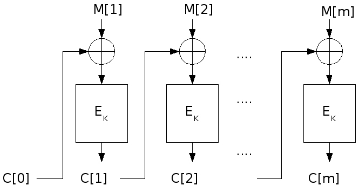


$A \rightarrow B$: Any A-secure scheme is B-secure
$A \not\rightarrow B$: There is an A-secure scheme that is not B-secure

Plain Encryption Does Not Provide Integrity

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

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

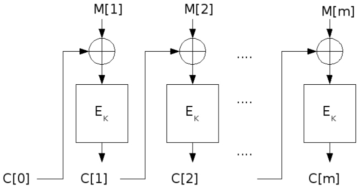


Question: Is CBC$ encryption INT-PTXT or INT-CTXT secure?

Plain Encryption Does Not Provide Integrity

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

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



Question: Is CBC$ encryption INT-PTXT or 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 $i = 0, \ldots, m$ do
$\quad C[i] \leftarrow \mathsf{E}_K(C[i-1] \oplus M[i])$
Return C

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

adversary A
$C[0]C[1]C[2] \xleftarrow{\$} \{0,1\}^{3n}$
$M[1]M[2] \leftarrow \textbf{Dec}(C[0]C[1]C[2])$

Then

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

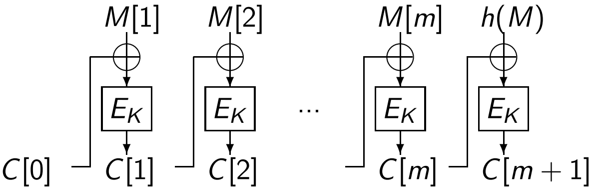
This violates INT-PTXT.

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

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

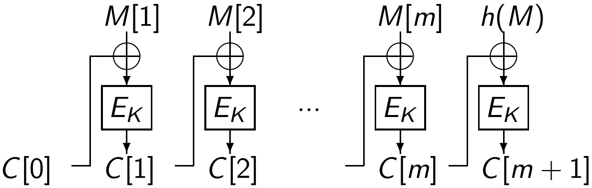


Here $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ is our block cipher and $h\colon \{0,1\}^* \to \{0,1\}^n$ is a "redundancy" function, for example

- $h(M[1] \ldots M[m]) = 0^n$
- $h(M[1] \ldots M[m]) = M[1] \oplus \cdots \oplus M[m]$
- 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

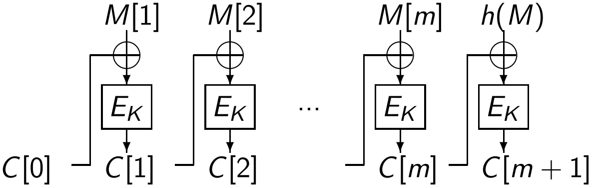


Let $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be our block cipher and $h\colon \{0,1\}^* \to \{0,1\}^n$ a redundancy function. Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}', \mathcal{D}')$ be CBC\$ encryption and define the encryption with redundancy scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E},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

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

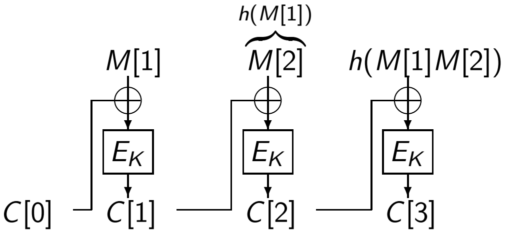
Encryption with Redundancy Fails

adversary A

$M[1] \xleftarrow{\$} \{0,1\}^n$; $M[2] \leftarrow h(M[1])$

$C[0]C[1]C[2]C[3] \xleftarrow{\$} \textbf{Enc}(M[1]M[2])$

$M[1] \leftarrow \textbf{Dec}(C[0]C[1]C[2])$



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 SUF-CMA MAC $\mathcal{MA}$[F] where $F : \{0,1\}^k \times \{0,1\}^* \to \{0,1\}^n$

	CBC$-AES	CTRC-AES	...
HMAC-SHA1			
CMAC			
PMAC			
UMAC			
⋮			

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 SUF-CMA MAC $\mathcal{MA}$[F] where $F : \{0,1\}^k \times \{0,1\}^* \rightarrow \{0,1\}^n$

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

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

Security	Achieved?
IND-CPA	
INT-PTXT	
INT-CTXT	

Encrypt-and-MAC

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

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

Security	Achieved?
IND-CPA	NO
INT-PTXT	
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

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-PTXT	
INT-CTXT	

Encrypt-and-MAC

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

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

Security	Achieved?
IND-CPA	NO
INT-PTXT	YES
INT-CTXT	

Why? F is a secure MAC and M is authenticated.

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-PTXT	YES
INT-CTXT	

Encrypt-and-MAC

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

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

Security	Achieved?
IND-CPA	
INT-PTXT	
INT-CTXT	

MAC-then-Encrypt

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

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

Security	Achieved?
IND-CPA	YES
INT-PTXT	
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 $\mathcal{E}_{K_e||K_m}(M)$
$T \leftarrow F_{K_m}(M)$
$C \xleftarrow{\$} \mathcal{E}'_{K_e}(M||T)$
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-PTXT	
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-PTXT	YES
INT-CTXT	

Why? F is a secure MAC and M is authenticated.

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

Security	Achieved?
IND-CPA	YES
INT-PTXT	
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 $\mathcal{E}_{K_e||K_m}(M)$
$C' \stackrel{\$}{\leftarrow} \mathcal{E}_{K_e}(M)$
$T \leftarrow F_{K_m}(C')$
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-PTXT	
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-PTXT	YES
INT-CTXT	

Why? If $\mathcal{D}_{K_e \| K_m}(C \| T)$ is new then C must be new too, so T must be a forgery.

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-PTXT	YES
INT-CTXT	

Encrypt-then-MAC

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

Alg $\mathcal{E}_{K_e\|\|K_m}(M)$	**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-PTXT	YES
INT-CTXT	YES

Why? If $\mathcal{D}_{K_e\|\|K_m}(C\|\|T)$ is new then

- If C is new, T must be a forgery
- If C is old, T is a strong forgery

Achieving IND-CCA

We saw that

$$\text{IND-CPA} + \text{INT-CTXT} \quad \Rightarrow \quad \text{IND-CCA}.$$

So an IND-CCA secure symmetric encryption scheme can be built as follows:

- Take any IND-CPA symmetric encryption scheme $\mathcal{SE}$
- Take any SUF-CMA MAC $\mathcal{MA}[\mathsf{F}]$
- Combine them in Encrypt-then-MAC composition

Example choices of the base primitives:

- $\mathcal{SE}$ is AES-CBC\$
- $\mathcal{MA}[\mathsf{F}]$ is AES-CMAC or HMAC-SHA1

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.

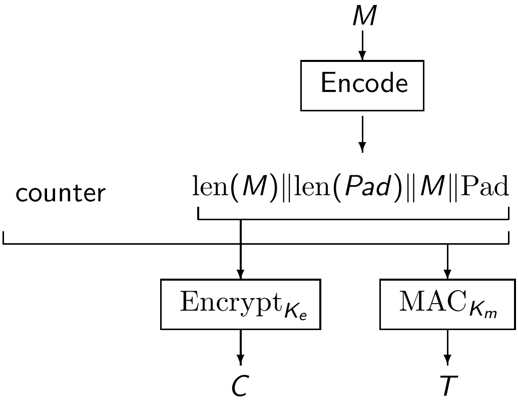
Generic Composition in Practice

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

AE in SSH



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, but became default only in 2009. Fixes also 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	Receiver
• $C \leftarrow \mathcal{E}_K(N, AD, M)$	• Receive (N, AD, C)
• Send (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 Privacy

Let $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme. Adversary is not allowed to repeat a nonce in its **LR** queries.

Game $\text{Left}_{\mathcal{AE}}$	Game $\text{Right}_{\mathcal{AE}}$
procedure Initialize $K \xleftarrow{\$} \mathcal{K}$	**procedure** Initialize $K \xleftarrow{\$} \mathcal{K}$
procedure LR(N, AD, M_0, M_1) Return $C \leftarrow \mathcal{E}_K(N, AD, M_0)$	**procedure LR**(N, AD, M_0, M_1) Return $C \leftarrow \mathcal{E}_K(N, AD, M_1)$

Associated to $\mathcal{AE}, A$ are the probabilities

$$\Pr\left[\text{Left}^A_{\mathcal{AE}} \Rightarrow 1\right] \qquad \Big| \qquad \Pr\left[\text{Right}^A_{\mathcal{AE}} \Rightarrow 1\right]$$

that A outputs 1 in each world. The (ind-cpa) advantage of A is

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}}(A) = \Pr\left[\text{Right}^A_{\mathcal{AE}} \Rightarrow 1\right] - \Pr\left[\text{Left}^A_{\mathcal{AE}} \Rightarrow 1\right]$$

AEAD Integrity

Let $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme. Adversary is not allowed to repeat a nonce in its **Enc** queries.

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

procedure Initialize
$K \xleftarrow{\$} \mathcal{K}$

procedure Enc(N, AD, M)
$C \leftarrow \mathcal{E}_K(N, AD, M)$
$S_{N,AD} \leftarrow S_{N,AD} \cup \{C\}$
return C

procedure Dec(N, AD, C)
$M \leftarrow \mathcal{D}_K(N, AD, C)$
if $(C \notin S_{N,AD} \wedge M \neq \perp)$ then
  win $\leftarrow$ true
return win

procedure Finalize
return win

The int-ctxt advantage of A is

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

AEAD Schemes

Generic composition: E&M, MtE, EtM extend and again EtM is the best.

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

Nonce-based symmetric encryption

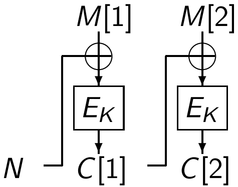
Worrying for the moment just about privacy, one could build a nonce-based symmetric encryption scheme by

- Using the nonce as IV in CBC mode
- Using the nonce as counter in CTR

Both are insecure, meaning fail to be IND-CPA, but can be fixed.

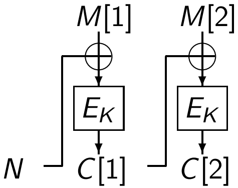
Nonce-based CBC encryption

Doesn't work:

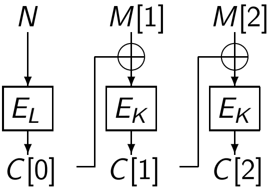


Nonce-based CBC encryption

Doesn't work:

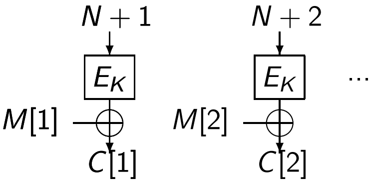


Works, and is easily justified under the assumption that E is a PRF:



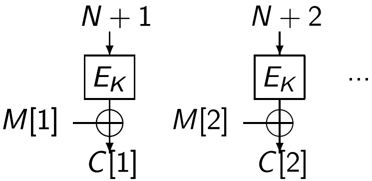
Nonce-based CTR encryption

Doesn't work:

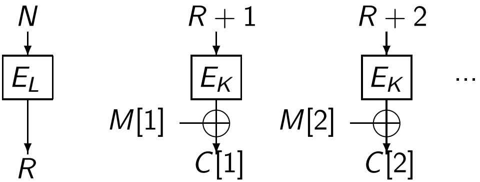


Nonce-based CTR encryption

Doesn't work:

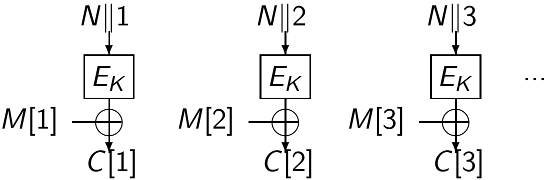


Works, and is easily justified under the assumption that E is a PRF:



Nonce-based CTR encryption

Also kind of works:



If maximum message length is 2^b blocks then nonce length is limited to $n - b$ bits.

We will see this tradeoff in some subsequent AEAD schemes.

Tweakable Block Ciphers [LRW]

A *tweakable block cipher* is a map

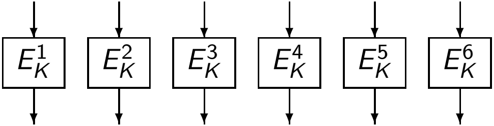
$$E\colon \{0,1\}^k \times \mathrm{TwSp} \times \{0,1\}^n \to \{0,1\}^n$$

such that

$$E_K^T\colon \{0,1\}^n \to \{0,1\}^n$$

is a permutation for every K, T, where $E_K^T(X) = E(K, T, X)$.

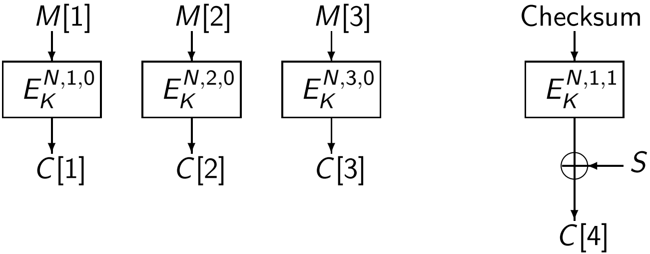
With a single key one thus implicitly has a large number of maps



These appear to be independent random permutations to an adversary who does not know the key K, even if it can choose the tweaks and inputs.

Tweakable block ciphers can be built cheaply from block ciphers [R].

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

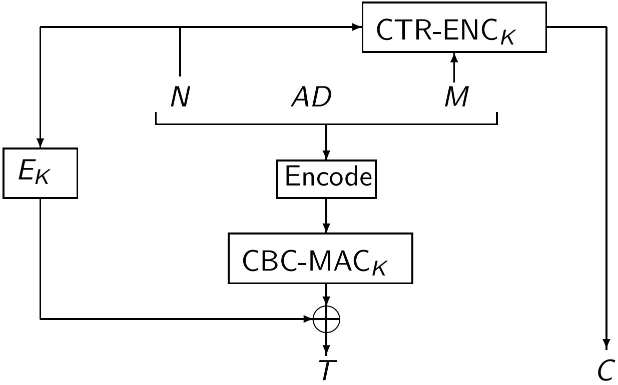
2-pass AEAD

- 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

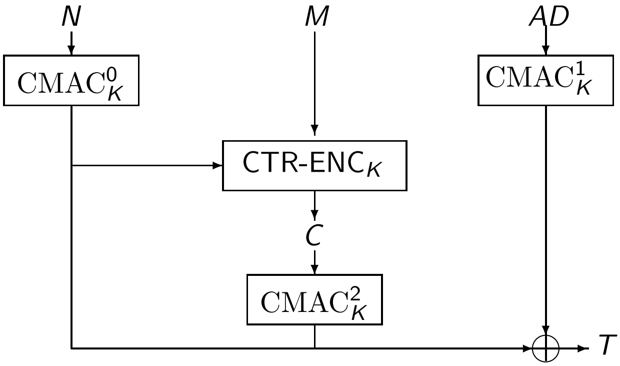
CTR-ENC is nonce-based counter mode encryption, and CBC-MAC is the basic CBC MAC. Ciphertext is $C \| T$

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]



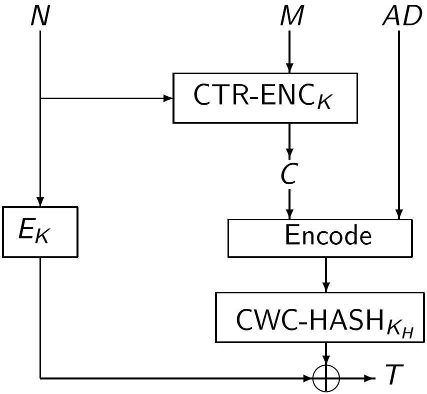
EtM-based but single key throughout

CTR-ENC is nonce-based counter mode encryption.

Online; can pre-process static AD; always 128-bit nonce; simple; same performance as CCM.

ANSI C12.22

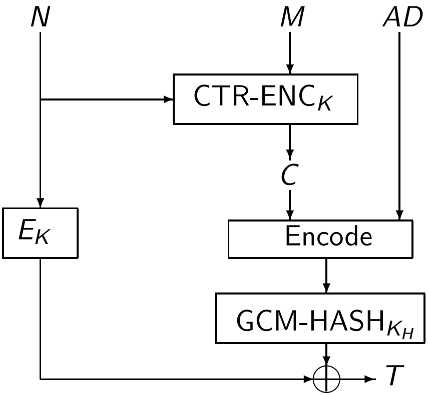
CWC [KVW]



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.

NIST SP 800-38D

Polynomial Hashes

Let F be a finite field. To data $C = C[0] \ldots C[m-1]$ with $C[i] \in F$ $(0 \leq i \leq m-1)$ we associate the polynomial

$$P_C(x) = \sum_{i=0}^{m-1} C[i] \cdot x^i$$

and let $H(K_H, C) = P_C(K_H)$. If $C_1 \neq C_2$, then for K_H chosen at random,

$$\begin{aligned} \Pr[H(K_H, C_1) = H(K_H, C_2)] &= \Pr[(P_{C_1} - P_{C_2})(K_H) = 0] \\ &\leq \frac{\max(m_1, m_2) - 1}{|F|}, \end{aligned}$$

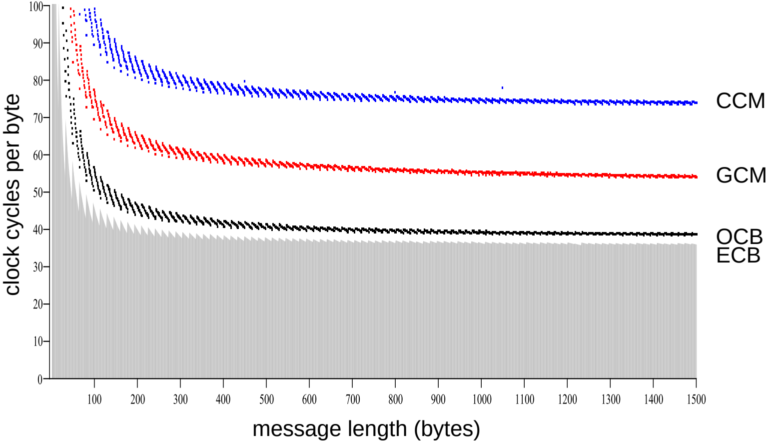
where m_i is the number of blocks in C_i.

CWC-HASH works over $F = \mathrm{GF}(p)$ where p is the prime $2^{127} - 1$, and is similar to Poly127 but is parallelizable. GCM-HASH works over $F = \mathrm{GF}(2^{128})$, which they argue is faster.

Critique of GCM [F]

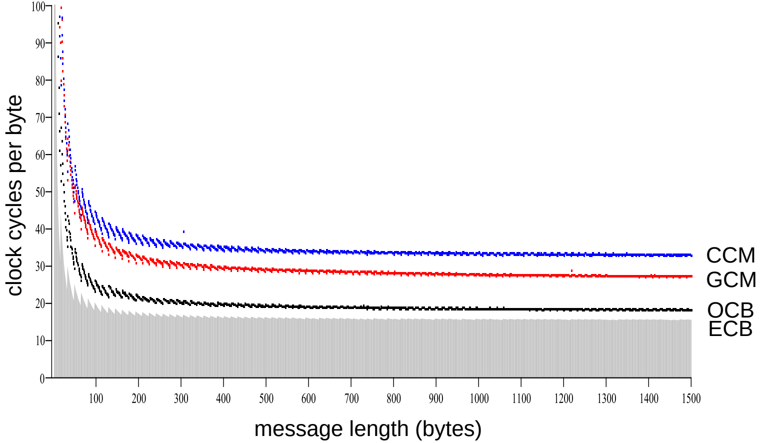
- Message length is at most $2^{36} - 64$ bytes which may not always be enough.
- Performance improvements require large per-key tables, which may be undesirable. (A wireless access point would need 1000 keys, hard for libraries to specifiy table sizes, tables contain confidential materials, etc.)
- As usual, forgery is possible via a birthday attack, but for some parameters the attacker can get the key.

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?