6: Robustness and Deterministic AE

Up to now we **developed** and **applied** security models by making certain constraints on the implementation:

- Chosen texts (ciphertexts, plaintexts, associated data)
  - → Eve must choose a full text (not **block-wise adaptive**)
- Stateful or randomized encryption
  - → Nonces never repeat (no **nonce-misuse**)
- IND-CCA security for AE:
  - → Never release unverified plaintexts (**RUP**); first check authenticity, then release plaintexts (or erease them without leaving a trace)!

In practice, some of these constraints may be (and too often actually is) violated. **Robustness** means a scheme provides a decent amount of security even if constraints are violated.

To prove certain forms of robustness, we have to tweak our security models (release Eve from some of the constraints).

Nonce Misuse

We know:

- Encryption must be stateful or randomized. Otherwise, Eve can discover certain things.
  (e.g., same message encrypted twice)
- But: what if messages never repeat anyways?
  (e.g., when encrypting secret keys)?
- What if we just happen to reuse the same nonce
  (or mishandle state or randomness, in general)?

Which security properties are still possible, and how good are the AE schemes we currently have at this?

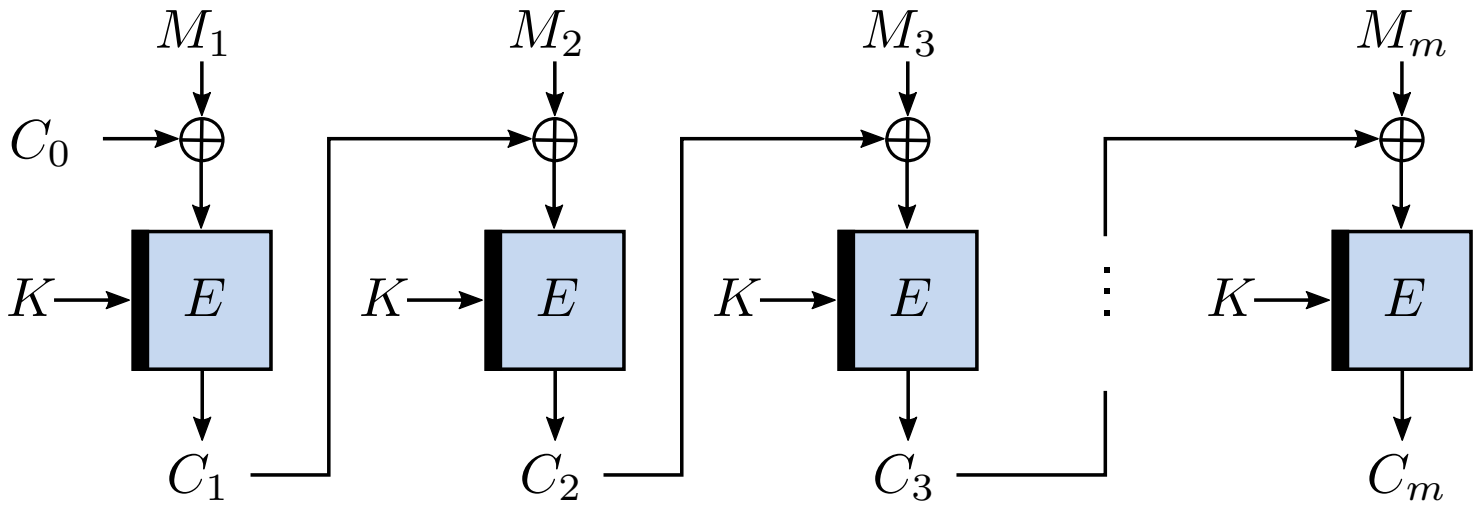
Reasons for Nonce Misuse

- Cryptography is hard to understand for application designers/implementors
- Environment may behave “unexpectedly”
  - System-clock reset (e.g., after changing the battery)
  - “Persistent” memory reset to older value (e.g., restoring damaged file from a backup)
  - Broken (hardware) random generator
  - Cloned virtual machine running the application
  - . . .
- Eve manages to trigger one of the above events on purpose

Block-Wise Adaptive Attacks

Example: CBC-encryption

- Practically relevevant for low-end systems
- Example: CBC-encryption:



  Eve will choose M_{i+1} after having seen C_i
- "Emulatable" by nonce misuse (asking more queries, though)

$\Rightarrow$ We will not specificially consider block-wise adaptive attacks

RUP

Release of Unverified Plaintexts

We learnt before:

Chosen-Plaintext-Security (IND-CPA)
+ Integrity of Ciphertexts (INT-CTXT)
= Chosen-Ciphertext-Security (IND-CCA)

Reason:

- Eve cannot benefit from chosen-ciphertext queries because she learns nothing from their decryption.

RUP bluntly contradicts the “because”.

RUP (2)

Reasons:

- Cryptography is hard to understand for application designers/implementors
- Temporary storage not overwritten
- Insufficient memory
- Demanding latency requirements
- ...

Workarounds:

- Output (parts of) the plaintext and hope for the best
- Decrypt-then-mask (Fouque et al, 2003, see blackboard)

A Mail from the Real Word

I have a question about the output of AEAD decryption. It is documented in RFC 5116 as follows:
It has only a single output, either a plaintext value P or a special symbol FAIL that indicates that the inputs are not authentic.

On decryption failure, if we not only output the special symbol FAIL but also the unauthenticated plaintext value P inadvertently, do you think this is a security vulnerability?

For example, an implementation of AES-GCM may perform GHASH and CTR-decryption simultaneously as the ciphertext comes in, so unauthenticated plaintext will be in the output buffer until we check final GHASH authentication tag. Is this a problem?

6.1: Misuse Attacks
Nonce Reuse – what to Expect?
Some reasonable (?) expectations

- Some plaintext information leaks:
  - Identical plaintexts
  - Common prefixes
  - etc.
- But not too much damage:
  1. Authentication not affected
  2. No immediate plaintext recovery

Nonce Reuse – what Really Happens!

A Double-Disaster for Almost All Current AE Schemes

1. Forgeries
2. Plaintext recoveries (often like “one-time-pad used twice”)

Attacks on On-Line Schemes

Parts: Fleischmann, Forler, Lucks 2012

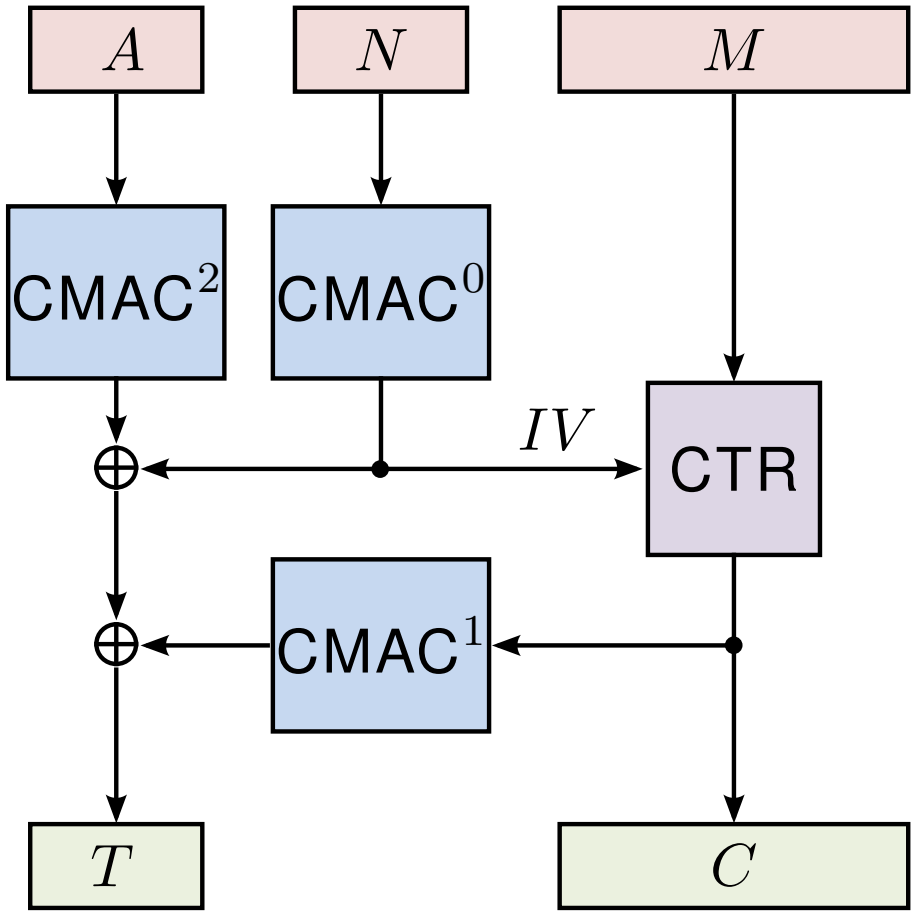
	Nonce Misuse		
Scheme	**Privacy**	**Integrity**	**RUP**
CCFB	$O(1)$	$O(1)$	$O(1)$
CHM	$O(1)$	$O(1)$	$O(1)$
COPA	n/a	n/a	$O(1)$
CWC	$O(1)$	$O(1)$	$O(1)$
EAX	$O(1)$	$O(1)$	$O(1)$
GCM	$O(1)$	$O(1)$	$O(1)$
IACBC	$O(1)$	$O(1)$	$O(1)$
IAPM	$O(1)$	$O(1)$	$O(1)$
OCB1	$O(1)$	$O(1)$	$O(1)$
OCB2	$O(1)$	$O(1)$	$O(1)$
OCB3	$O(1)$	$O(1)$	$O(1)$
RPC	$O(1)$	$O(1)$	$O(1)$
TAE	$O(1)$	$O(1)$	$O(1)$
XCBC-XOR	$O(2^{n/4})$	$O(1)$	$O(1)$

Attacks on Off-Line Schemes

Parts: Fleischmann, Forler, Lucks 2012

	Nonce Misuse		
Scheme	**Privacy**	**Integrity**	**RUP**
BTM	n/a	n/a	$O(1)$
CCM	$O(1)$	$\ll 2^{(n/2)}$	$O(1)$
HBS	n/a	n/a	$O(1)$
SIV	n/a	n/a	$O(1)$

Recall EAX



Misusing EAX

- Privacy under NM: Negl. effort
  (same key stream used twice)
- Authenticity under NM: Negl. effort

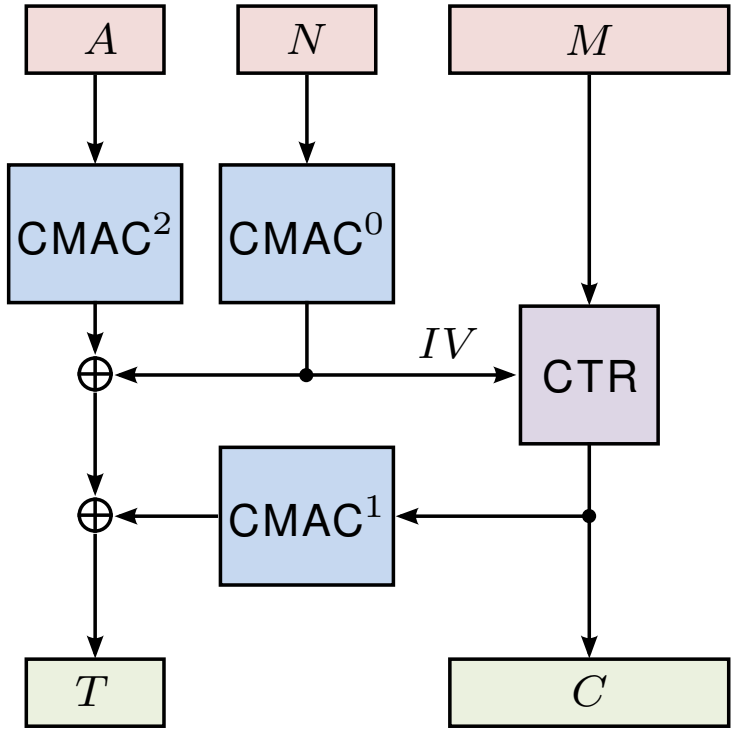
$$(C, T) \leftarrow \mathsf{EAX}(N, M, A)$$
$$(C, T') \leftarrow \mathsf{EAX}(N, M, A')$$
$$(C', T'') \leftarrow \mathsf{EAX}(N', M', A)$$

Forgery:

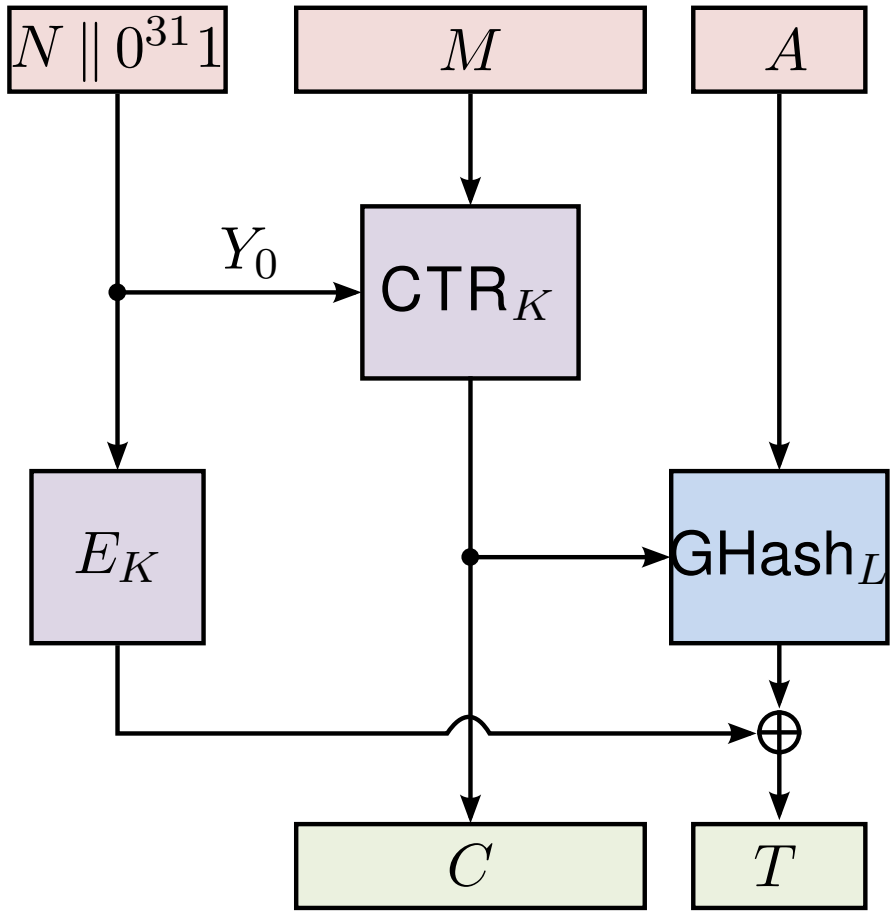
$$(C', T''') \leftarrow \mathsf{EAX}(N', M', A')$$

with $T''' = T'' \oplus T' \oplus T$

- Privacy under RUP: Negl. effort
  (easy to find key stream)



Recall GCM



Misusing GCM

- Privacy under NM: Negl. effort
  (same key stream used twice)
- Authenticity under NM: Negl. effort

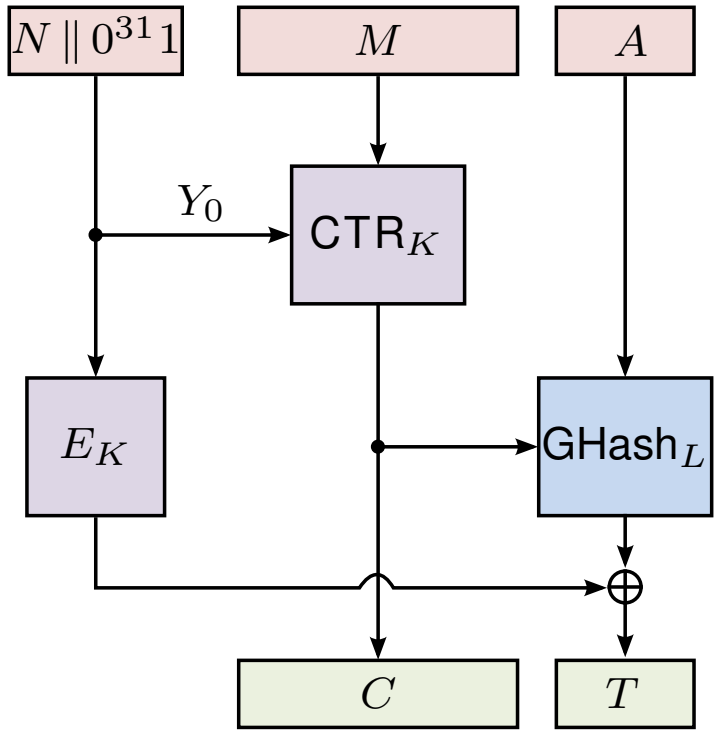
$$(C, T) \leftarrow \mathsf{GCM}(N, M, A)$$
$$(C, T') \leftarrow \mathsf{GCM}(N, M', A')$$
$$(C', T'') \leftarrow \mathsf{GCM}(N', M, A)$$

Forgery:

$$(C', T''') \leftarrow \mathsf{GCM}(N', M', A')$$

with $T''' = T'' \oplus T' \oplus T$.

- Moreover, the GHash key L can be compromised ($\rightarrow$ next slide)
- Privacy under RUP: Negl. effort
  (easy to find key stream)



Searching for $L = E_K(0)$ (under NM)

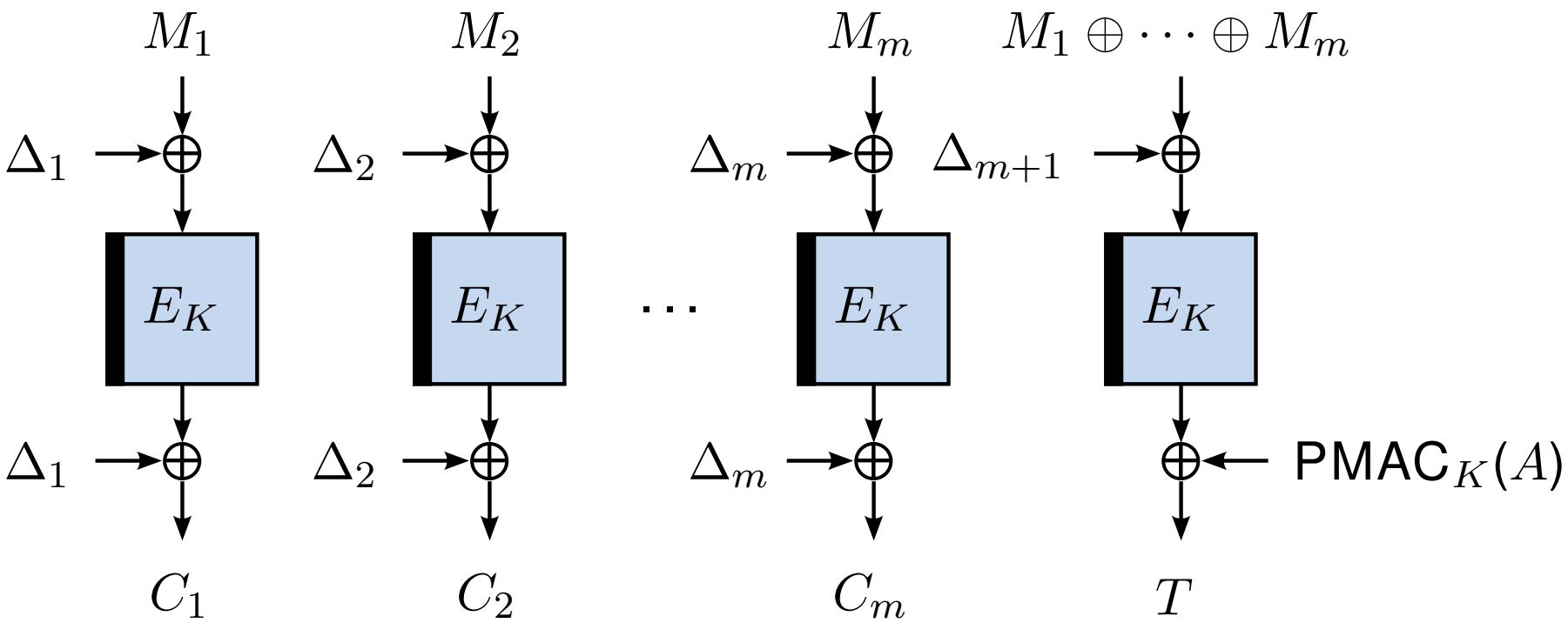
Same N, same Y_0. Given two **"Tags"** T_X and T_Y, compute L = $\mathbf{E_k(0)}$ as root of a polynomial!

$$\overbrace{(LX_1 \oplus L^2 X_2 \oplus \cdots \oplus L^\ell X_\ell)}^{\mathsf{GHash}_L(X)} \oplus \mathbf{E_K(Y_0)} \oplus T_X = 0$$

$$\overbrace{(LY_1 \oplus L^2 Y_2 \oplus \cdots \oplus L^\ell Y_\ell)}^{\mathsf{GHash}_L(Y)} \oplus \mathbf{E_K(Y_0)} \oplus T_Y = 0$$

$$\overbrace{(L(X_1 \oplus Y_1) \oplus L^2(X_2 \oplus Y_2) \oplus \cdots \oplus L^\ell(X_\ell \oplus Y_\ell) \oplus T_1 \oplus T_2)}^{\mathsf{GHash}(X \oplus Y) \oplus T_1 \oplus T_2} = 0$$

Recall OCB


M_1
M_2
M_m
$M_1 \oplus \cdots \oplus M_m$
Δ_1
Δ_2
Δ_m
Δ_{m+1}
E_K
E_K
$\cdots$
E_K
E_K
Δ_1
Δ_2
Δ_m
$\mathsf{PMAC}_K(A)$
C_1
C_2
C_m
T

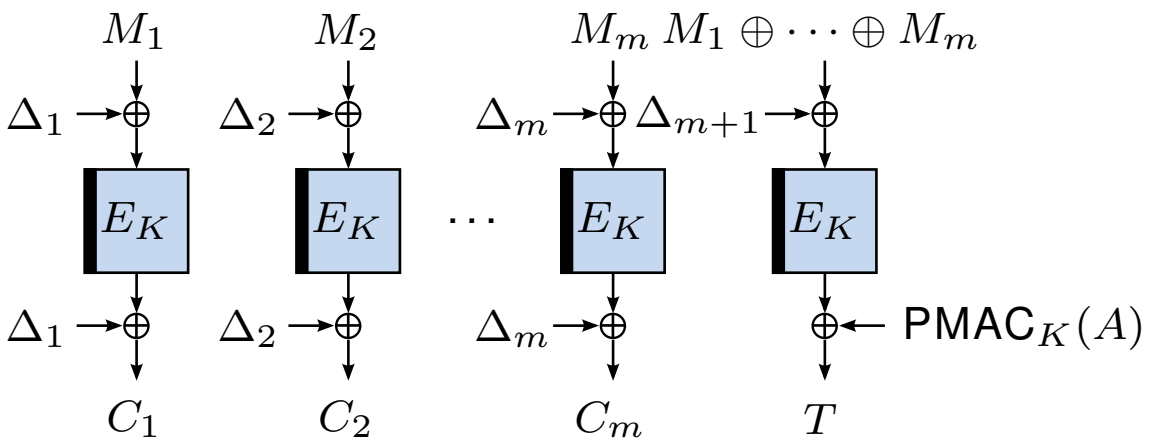
Misusing OCB

- Privacy under NM: Low (ECB-like, Eve sees if $M_i = M'_i$)
- Authenticity under NM: Negl. effort

$$(C, T) \leftarrow \mathsf{OCB}(N, M, A)$$
$$(C, T') \leftarrow \mathsf{OCB}(N, M, A')$$
$$(C', T'') \leftarrow \mathsf{OCB}(N', M', A)$$

Forgery: $(C', T''') \leftarrow \mathsf{OCB}(N', M', A')$, with $T''' = T'' \oplus T' \oplus T$.

- Privacy under RUP: Low (ECB-like)



First AEAD Generation

very efficient, does the job, . . .

. . . but is dangerously brittle!

6.2: Deterministic Authenticated Encryption (DAE)

Rogaway, Shrimpton 2006

Best of both worlds:

- **Full security** (INT-CTXT + RoR-CPA, **nonce-respecting** Eve)

  (Nonce is now included in A)
- **Best achievable security** even if Eve **misuses nonces**:
  - Consider: $(A, M) \to C$, and $(A', M') \to C'$.
  - If $(A, M) = (A', M') \implies C = C'$ (unavoidable!)
  - But if $A \neq A' \implies C$ and C' are indistinguishable from independent random variables, regardless of $M = M'$ or $M \neq M'$

DAE Definition

Deterministic Authenticated Encryption with Associated Data

- Sets:
  - MESSAGES $\subseteq \{0,1\}^*$
  - KEYS $\subseteq \{0,1\}^k$
  - CIPHERTEXTS $\subseteq \{0,1\}^*$
  - HEADERS $\subseteq \{0,1\}^*$
- Functions:
  - Gen: $\ldots \to$ KEYS
  - Encr: KEYS $\times$ HEADERS $\times$ MESSAGES $\to$ CIPHERTEXTS
  - Decr: KEYS $\times$ HEADERS $\times$ CIPHERTEXTS $\to$ MESSAGES $\cup \perp$
  - Written as $\mathsf{Encr}_K(\ldots)$, $\mathsf{Decr}_K(\ldots)$
- $\forall K \in$ KEYS, $H \in$ HEADERS, $M \in$ MESSAGES :

$$\mathsf{Decr}_K(H, \mathsf{Encr}_K(H, M)) = M.$$

The DAE Experiment

Definition 41 (The DAE Experiment)

1. The oracle chooses $K \xleftarrow{\$} \text{KEYS}$ and tosses a fair coin $\beta \xleftarrow{\$} \{0,1\}$.
2. For each i in $\{1,\ldots,q\}$, Eve makes either of two queries to the oracle:

   $$\begin{aligned} \text{Encryption} &: \text{HEADERS} \times \text{MESSAGES} \to \text{CIPHERTEXTS} \\ \text{Decryption} &: \text{HEADERS} \times \text{CIPHERTEXTS} \to \text{MESSAGES} \cup \perp \end{aligned}$$

   - If $\beta = 1$, the oracle applies Encr_K and Decr_K to Eve's queries.
   - Otherwise, the oracle responds with random values of the appropriate length to encryption queries, and with "$\perp$" to all decryption queries.

   $(*)$ Eve omits all queries, for which she would already know the answer in the "real" case.
3. Eve outputs a guess $\beta' \in \{0,1\}$. She wins iff $\beta' = \beta$.

Attack Scenario (2)

Definition 42 (DAE Advantage)

Let $\Pi = (\textsc{Encr}, \textsc{Decr})$ be an deterministic AE scheme. Let $K \xleftarrow{\$} \textsc{Keys}$. Let $\mathcal{A}$ be a DAE adversary on Π. Then, $\mathcal{A}$'s *DAE advantage* is defined as

$$\mathbf{Adv}^{\mathsf{DAE}}_{\Pi}(\mathcal{A}) := \left|\Pr\left[\mathcal{A}^{\textsc{Encr}_K(\cdot,\cdot),\textsc{Decr}_K(\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$(\cdot,\cdot),\perp(\cdot,\cdot)} \Rightarrow 1\right]\right|$$

The condition $(*)$ means that

- Eve never repeats an encryption query
- nor does she ever ask for the decryption of (H, C) if C was the answer of an encryption query (H, M).
- Everything else is allowed. E.g., if C and C' are answers on encryption queries (H, M) and (H', M') and $H \neq H'$, Eve can ask for the decryption of (H, C') and (H', C).

Comparison

DAE advantage $\Leftrightarrow$ RoR-CPA advantage $+$ Int-CTXT advantage

Consider adversaries Eve, Eve′ und Eve″, i.e., efficient attackers:

- Eve attacks with DAE advantage a,
- Eve′ attacks with RoR-CPA advantage b and
- Eve″ forges with Int-CTXT advantage c.

Questions:

1. Let a be significant. Can you show that Eve' or Eve" with significant b or c exist? And how large would b and c actually become?
2. Let b be significant. ...
3. Let c be significant. ...

Comparison (2)

Theorem 43

1. $\boxed{b + qc \geq a}$ *Let Eve with DAE advantage* a *be given. Then there exist Eve′ and Eve″ being at least as efficient as Eve with* RoR-CPA *advantage* b *and* Int-CTXT *advantage* c*, such that*

$$b + qc \geq a.$$

2. $\boxed{a \geq b}$ *If Eve′ achieves the* RoR-CPA *advantage* b*, she also achieves the same DAE advantage* $a = b$*.*

3. $\boxed{a \geq c}$ *If Eve″ forges ciphertexts with probability* c*, she also achieves the DAE advantage* $a = c$*.*

6.3: Robust Schemes

Maximize Security under Nonce Reuse (and RUP)

- **If plaintexts** equal and **AD/N** equal
  then same **tag**, same **ciphertext**

- **If plaintexts** different or **AD/N** different
  then random **tag**, random **ciphertext**

Nonce-Reuse

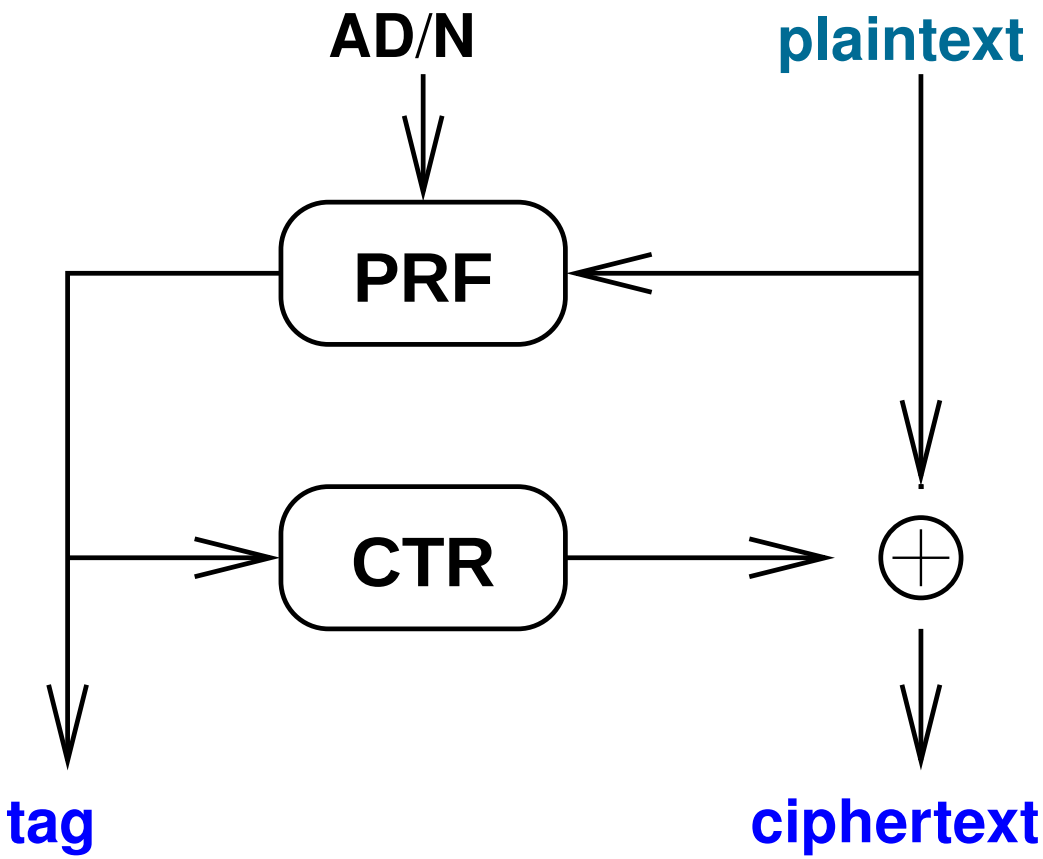
Rogaway, Shrimpton, 06: "Deterministic AE", SIV

Nonce-Reuse und RUP:

Abed, Forler, List, Lucks, Wenzel, 16 RIV

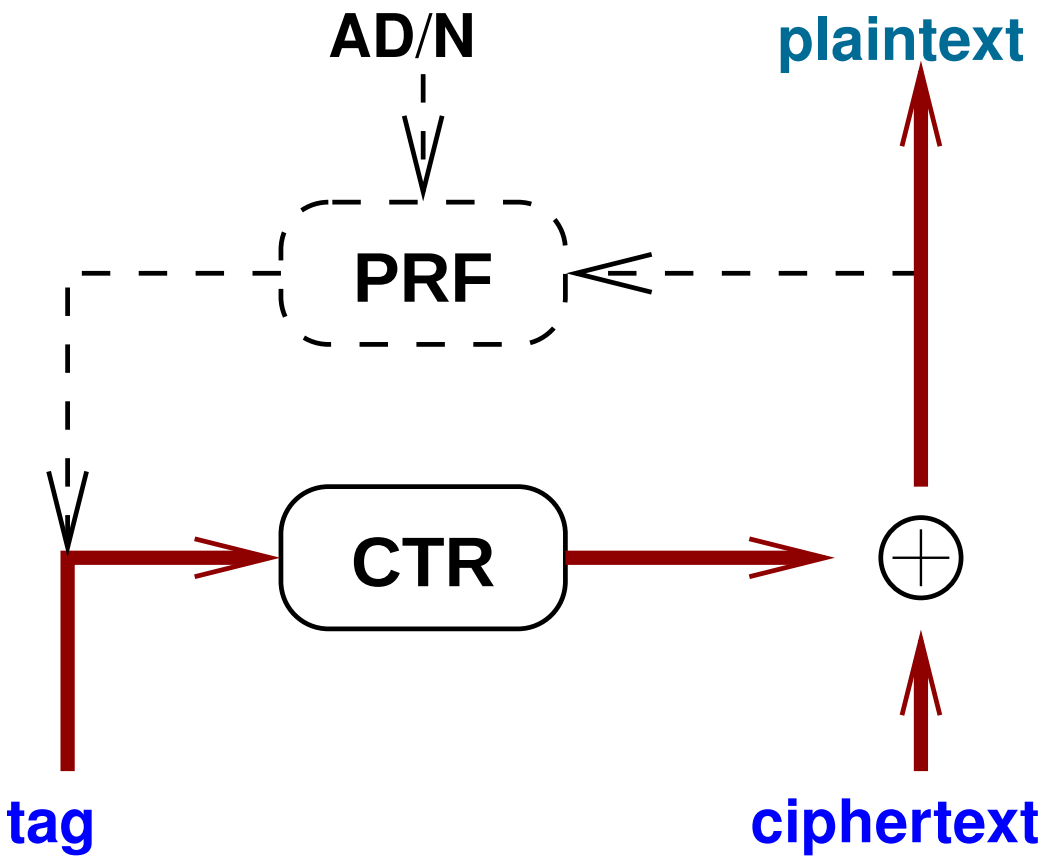
SIV: Deterministic Authenticated Encryption

"Proof Sketch": For CP Queries, **tag** Depends on All Inputs

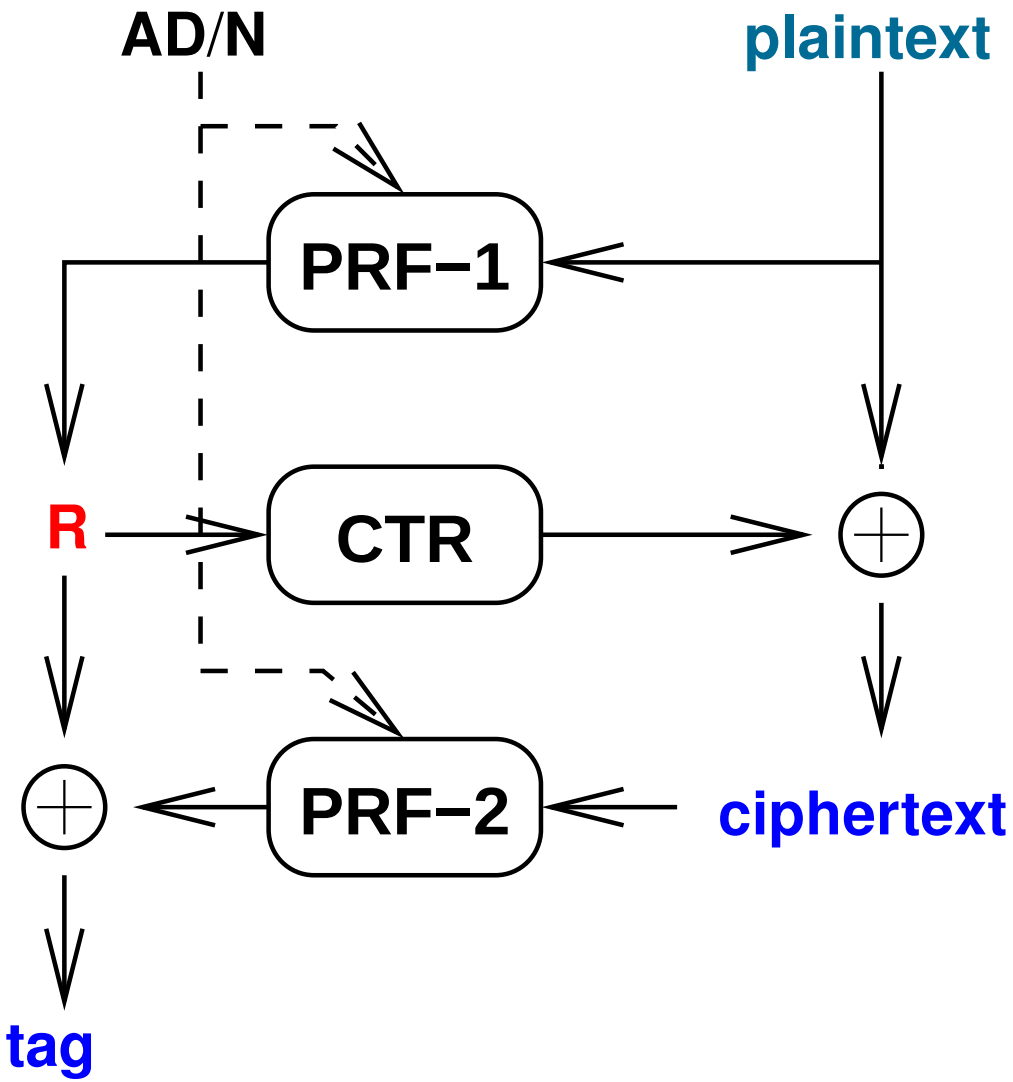


SIV: Insecure under RUP

"Chosen-Ciphertext RUP": Same **tag**, Same Keystream CTR(**tag**)

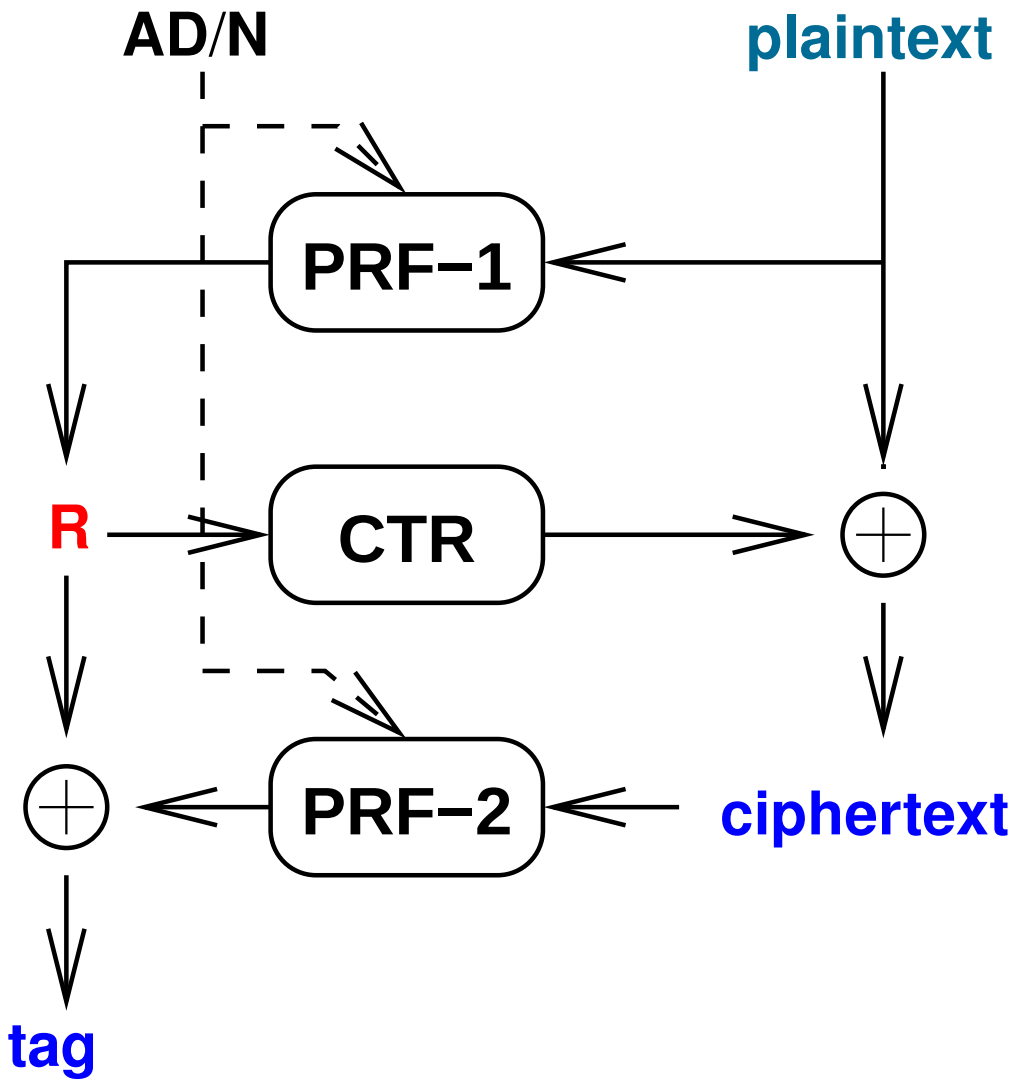


RIV: SIV with An Additional Round



RIV: Also Secure Under RUP!

"Proof Sketch": **R** Depends on All Inputs, for Both CP and CC Queries



Instantiation of RIV

- Based on AES-128
- PRFs: Encode-Hash-Encrypt:
  - Unique encoding for inputs
  - Apply CLHASH, a multi-stage universal hash function (largish internal key; sizes 256 bytes and 1024 bytes)
  - Feed result into block cipher
- Encryption: AES in CTR mode

Performance of RIV and POET on Haswell

POET: on-line cipher.