

Towards Optimally Secure Deterministic Authenticated Encryption Schemes

soft merge with

Making GCM Great Again: Toward Full Security and Longer Nonces

Ashwin Jha

RUB

Byeonghak Lee

Samsung SDS

Eurocrypt 2025

6 May, 2025

Towards Optimally Secure Deterministic Authenticated Encryption Schemes

Yu Long Chen

KU Leuven

Avijit Dutta

TCG CREST

Ashwin Jha

RUB

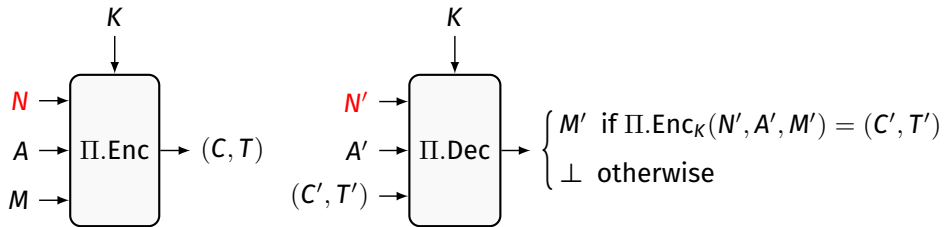
Mridul Nandi

ISI Kolkata

Eurocrypt 2025

6 May, 2025

Authenticated Encryption with Associated Data

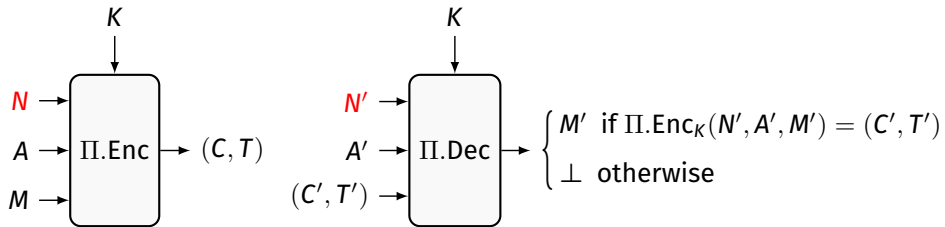


- AEAD *encrypts* the message M + *authenticates* the metadata & message (A, M)
- Widely deployed (TLS, IPsec, wireless standards)

GCM CCM ChaCha20-Poly1305 Ascon

- Nonce is supposed to be unique in encryption

Authenticated Encryption with Associated Data

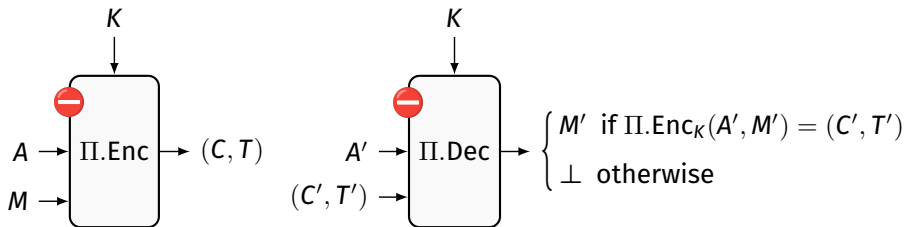


- AEAD *encrypts* the message M + *authenticates* the metadata & message (A, M)
- Widely deployed (TLS, IPsec, wireless standards)

GCM CCM ChaCha20-Poly1305 Ascon

- Nonce is **supposed to be unique** in encryption

Deterministic AEAD [RS: EC '06]



- AEAD *without* a nonce [can be absorbed in the associated data]
- Encryption at rest (iCloud, AWS) and tokenization (PCI-compliant systems)

SIV GCM-SIV

Why Use a Nonce?

Uniqueness of nonce in encryption ensures security and efficiency

- Security:
 - DAEAD *leaks equality* when message + metadata repeat.
 - Nonce ensures *fresh* randomness per encryption query
- Efficiency:
 - DAEAD are inherently *two-pass* (rate¹ is capped at 0.5)
 - Nonce allows for *single-pass* schemes

¹The ratio of number of n -bit blocks in the input to the number of primitive calls.

Why Use a Nonce?

Uniqueness of nonce in encryption ensures security and efficiency

- Security:
 - DAEAD *leaks equality* when message + metadata repeat.
 - Nonce ensures *fresh* randomness per encryption query
- Efficiency:
 - DAEAD are inherently *two-pass* (rate¹ is capped at 0.5)
 - Nonce allows for *single-pass* schemes

¹The ratio of number of n -bit blocks in the input to the number of primitive calls.

Why Use a Nonce?

Uniqueness of nonce in encryption ensures security and efficiency

- Security:
 - DAEAD *leaks equality* when message + metadata repeat.
 - Nonce ensures *fresh* randomness per encryption query
- Efficiency:
 - DAEAD are inherently *two-pass* (rate¹ is capped at 0.5)
 - Nonce allows for *single-pass* schemes



Nonce-reuse is *strictly* prohibited!

¹The ratio of number of n -bit blocks in the input to the number of primitive calls.

The Curse of Nonce-Misuse

The Curse of Nonce-Misuse

Here Come The ☯ Ninjas

🚩 CVE-2017-3225 Detail

MODIFIED

This CVE record has been updated after NVD enrichment efforts were completed. Enrichment data supplied by the NVD may require amendment due to these changes.

Description

Das U-Boot is a device bootloader that can read its configuration from an AES encrypted file. For devices utilizing this environment mode, U-Boot's use of a zero initialization vector may allow attacks against the underlying cryptographic implementation and allow an attacker to decrypt the data. Das U-Boot's AES-CBC encryption feature uses a zero (0) initialization vector. This allows an attacker to perform dictionary attacks on encrypted data produced by Das U-Boot to learn information about the encrypted data.

The secure socket
SSL standard allo-
cations usually utilize

Description

The `__construct` function in `Framework/Encryption/Crypt.php` in Magento 2 uses the PHP `rand` function to generate an initialization vector, which makes it easier for remote attackers to defeat cryptographic protection mechanisms.

On the Security of RC4 in TLS¹

Nadhem J. AlFardan
Information Security Group,
Royal Holloway, University of London

Daniel J. Bernstein
University of Illinois at Chicago and
Technische Universiteit Eindhoven

Kenneth G. Paterson
Information Security
Royal Holloway, University of London

🚩 CVE-2020-5408 Detail

MODIFIED

This CVE record has been updated after NVD enrichment efforts were completed. Enrichment data supplied by the NVD may require amendment due to these changes.

Description

Spring Security versions 5.3.x prior to 5.3.2, 5.2.x prior to 5.2.4, 5.1.x prior to 5.1.10, 5.0.x prior to 5.0.16 and 4.2.x prior to 4.2.16 use a fixed null initialization vector with CBC Mode in the implementation of the queryable text encryptor. A malicious user with access to the data that has been encrypted using such an encryptor may be able to derive the unencrypted values using a dictionary attack.

Lucky Thirteen: Breaking the TLS and DTLS Record Protocols

🚩 CVE-2014-5386 Detail

DEFERRED

This CVE record is not being prioritized for NVD enrichment efforts due to resource or other concerns.

Description

The `mcrypt_create_iv` function in `hhp/runtime/ext/mcrypt/ext_mcrypt.cpp` in Facebook HipHop Virtual Machine (HHVM) before 3.3.0 does not seed the random number generator, which makes it easier for remote attackers to defeat cryptographic protection mechanisms by leveraging the use of a single initialization vector.

tion-layer
handshaker
h, session
l the TLS
manage-

This POODLE Bites: Exploiting The SSL 3.0 Fallback

Security Advisory

🚩 CVE-2011-3389 Detail

DEFERRED

This CVE record is not being prioritized for NVD enrichment efforts due to resource or other concerns.

Description

The SSL protocol, as used in certain configurations in Microsoft Windows and Microsoft Internet Explorer, Mozilla Firefox, Google Chrome, Opera, and other products, encrypts data by using CBC mode with chained initialization vectors, which allows man-in-the-middle attackers to obtain plaintext HTTP headers via a blockwise chosen-boundary attack (BCBA) on an HTTPS session, in conjunction with JavaScript code that uses (1) the HTML5 WebSocket API, (2) the Java URLConnection API, or (3) the Silverlight WebClient API, aka a "BEAST" attack.

replaced by its successors TLS 1.0 [RFC2246], TLS 1.1 [RFC4346],

ACM
message in
case.

The Curse of Nonce-Misuse

Here Come The ☯ Ninjas

Lucky Thirteen: Breaking the TLS and DTLS Record Protocols

CVE-2013-0056

MODIFIED

This CVE record has been updated after NVD enrichment efforts were completed. Enrichment data supplied by the NVD may require amendment due to these changes.

Description

Das U-Boot is a mode, U-Boot's attacker to decrypt dictionary attack

SS

tion

Authentication Failures in NIST version of GCM

Antoine Joux

DGA

and

Université de Versailles St-Quentin-en-Yvelines

PRISM

45, avenue des Etats-Unis

78035 Versailles Cedex, France

Antoine.Joux@m4x.org

Abstract. In this note, we study the security of the Galois/Counter mode authenticated encryption recently published by NIST. We show how an adversary can recover the secret key of the keyed hash function underlying the authentication, using a chosen IV attack. Once this secret

MODIFIED

This CVE record has been updated after NVD enrichment efforts were completed. Enrichment data supplied by the NVD may require amendment due to these changes.

Description

Spring Security versions 5.3.x prior to 5.3.2, 5.2.x prior to 5.2.4, 5.1.x prior to 5.1.10, 5.0.x prior to 5.0.16 and 4.2.x prior to 4.2.16 use a fixed null initialization vector with CBC Mode in the implementation of the queryable text encryptor. A malicious user with access to the data that has been encrypted using such an encryptor may be able to derive the unencrypted values using a dictionary attack.

Description

The SSL protocol, as used in certain configurations in Microsoft Windows and Microsoft Internet Explorer, Mozilla Firefox, Google Chrome, Opera, and other products, encrypts data by using CBC mode with chained initialization vectors, which allows man-in-the-middle attackers to obtain plaintext HTTP headers via a blockwise chosen-boundary attack (BCBA) on an HTTPS session, in conjunction with JavaScript code that uses (1) the HTML5 WebSocket API, (2) the Java URLConnection API, or (3) the Silverlight WebClient API, aka a "BEAST" attack.

replaced by its successors TLS 1.0 [RFC2246], TLS 1.1 [RFC4346],

The Curse of Nonce-Misuse

Here Come The \oplus Ninjas

Lucky Thirteen: Breaking the TLS and DTLS Record Protocols

CVE-2017-20052

MODIFIER

This CVE recommendation

Descript

Das U-Boot is a mode, U-Boot's attacker to dec dictionary atta

SS
lie

Univer

III-4

MODIFIED

This CVE record has been updated after NVD enrichment amendment due to these changes.

Description

Spring Security versions 5.3.x prior to 5.3.4, 5.2.x prior to 5.2.10, and 5.1.x prior to 5.1.10 were vulnerable to a dictionary attack. The initialization vector with CBC Mode in the implementation of the queryable text encryption. An attacker with access to the encrypted data can be able to derive the unencrypted values using a dictionary attack.

Authentication Failures in NIST version of GCM

Non-Disrespecting Adversaries: Practical Forgery Attacks on GCM in TLS

Hanno Böck¹, Aaron Zauner², Sean Devlin³, Juraj Somorovsky⁴ and Philipp Jovanovic⁵

¹<https://hboeck.de>, hanno@hboeck.de
auner@sba-research.at

²SBA Research gGmbH, azauner@sba-research.org, lambda: resilient.systems, azet@azet.org
³Independent, seanpatrickdevlin@gmail.com
⁴Horst Görtz Institute for IT Security, Ruhr University
⁵École Polytechnique

³Independent, seanpatrickdevlin@gmail.com

⁵École Polytechnique Fédérale de Lausanne (EPFL), philipp.jovanovic@epfl.ch

École Polytechnique Fédérale de Lausanne (EPFL), philipp.jovanovic@epfl.ch

Abstract

Abstract

We investigate nonce reuse issues with the GCM block cipher mode as used in TLS and focus in particular on

POODLE-TLS [23] (implementation bugs). All those attacks did not exploit weaknesses of CBC per se, but took advantage of the particular way how CBC was de-

Need for Beyond-the-Birthday Bound Security

- GCM, CCM, and OCB[†] are *limited* to birthday bound security
AES-{GCM,CCM,OCB[†]} is secure up to 2^{64} queries
- 64-bit security might be *insufficient*
 - exabyte-scale ($\simeq 2^{60}$) in use, zetabyte-scale ($\simeq 2^{70}$) expected
 - Limited *generic* multi-user security
- Standardise a bigger block cipher [an effective long term solution(?)]
 - Replacing AES-128 might not be viable
 - Noticeable *setup time* expected
- BBB secure (nonce-based) AEAD modes
 - CHM: full n -bit security
 - SCM: *graceful degradation (limited to $n/2$ -bit security for arbitrary misuse)*
 - SIV_r: BBB nonce-misuse security (*highly inefficient*)

Need for Beyond-the-Birthday Bound Security

- GCM, CCM, and OCB[†] are *limited* to birthday bound security
AES-{GCM,CCM,OCB[†]} is secure up to 2^{64} queries
- 64-bit security might be *insufficient*
 - exabyte-scale ($\simeq 2^{60}$) in use, zetabyte-scale ($\simeq 2^{70}$) expected
 - Limited *generic* multi-user security
- Standardise a bigger block cipher [an effective long term solution(?)]
 - Replacing AES-128 might not be viable
 - Noticeable **setup time** expected [hardware support, general confidence]
- BBB secure (nonce-based) AEAD modes
 - CHM: full n -bit security
 - SCM: graceful degradation (*limited to $n/2$ -bit security for arbitrary misuse*)
 - SIV_r: BBB nonce-misuse security (*highly inefficient*)

Need for Beyond-the-Birthday Bound Security

- GCM, CCM, and OCB[†] are *limited* to birthday bound security
AES-{GCM,CCM,OCB[†]} is secure up to 2^{64} queries
- 64-bit security might be *insufficient*
 - exabyte-scale ($\simeq 2^{60}$) in use, zetabyte-scale ($\simeq 2^{70}$) expected
 - Limited *generic* multi-user security
- Standardise a bigger block cipher [an effective long term solution(?)]
 - Replacing AES-128 might not be viable
 - Noticeable **setup time** expected
- BBB secure (nonce-based) AEAD modes [CHM, SCM, SIV_r, GCM-SIV, Θ CB, Romulus, LightOCB]
 - CHM: full n -bit security
 - SCM: *graceful* degradation (*limited to $n/2$ -bit security for arbitrary misuse*)
 - SIV_r: BBB nonce-misuse security (*highly inefficient*)

Need for Beyond-the-Birthday Bound Security

- GCM, CCM, and OCB[†] are *limited* to birthday bound security
AES-{GCM,CCM,OCB[†]} is secure up to 2^{64} queries
- 64-bit security might be *insufficient*
 - exabyte-scale ($\simeq 2^{60}$) in use, zetabyte-scale ($\simeq 2^{70}$) expected
 - Limited *generic* multi-user security
- Standardise a bigger block cipher [an effective long term solution(?)]
 - Replacing AES-128 might not be viable
 - Noticeable **setup time** expected
- BBB secure (nonce-based) AEAD modes
 - CHM: full n -bit security (*insecure with a single misuse*)
 - SCM: *graceful* degradation (*limited to $n/2$ -bit security for arbitrary misuse*)
 - SIV_r: BBB nonce-misuse security (*highly inefficient*)

The Goal

To solve two problems:

- *Unique* nonce requirement
- *Limited* security (birthday bound)

The Goal

To solve two problems:

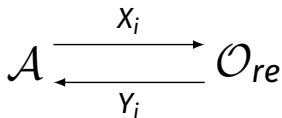
- *Unique* nonce requirement
- *Limited* security (birthday bound)

*Design a block cipher-based efficient,
misuse-resistant BBB-secure AEAD mode*

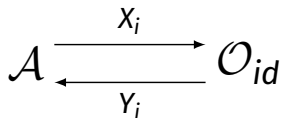
Security Notions: A Quick Recap

The Distinguishing Game

Real World (**RW**)



Ideal World (**IW**)



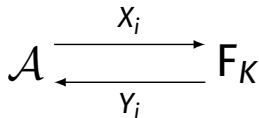
$$\mathbf{Adv}_{\mathcal{O}_{re}}^{\text{game}}(\mathcal{A}) := \left| \Pr(\mathcal{A} \text{ returns } 1 \text{ in } \mathbf{RW}) - \Pr(\mathcal{A} \text{ returns } 1 \text{ in } \mathbf{IW}) \right|$$

- Adversary's resources: q (query), ℓ (max. length), σ (total data) etc.
- Game: ideal world *functionality* + adversary's *power*

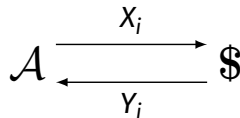
Security Notions: A Quick Recap

Pseudorandom Function (PRF)

RW



IW

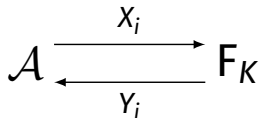


- Ideal world: a uniform random function $\$$
- \mathcal{A} makes *chosen* plaintext queries
- $\text{Adv}_F^{\text{prf}}(\mathcal{A})$: the PRF advantage of \mathcal{A} against F

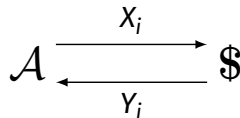
Security Notions: A Quick Recap

Random IV-based PRF (\$-PRF)

RW



IW

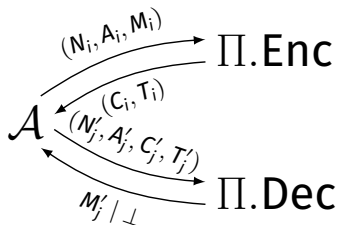


- Ideal world: a uniform random function $\$$
- \mathcal{A} makes **random** plaintext queries
- $\text{Adv}_F^{\$, \text{prf}}(\mathcal{A})$: the $\$$ -PRF advantage of \mathcal{A} against F

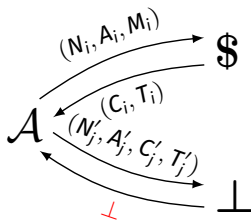
Security Notions: A Quick Recap

Misuse-resistant AE (MRAE)

RW



IW

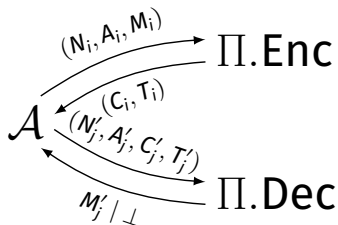


- Ideal world: a uniform random function $\$$ and the *reject* oracle \perp
- \mathcal{A} 's queries must satisfy $(N'_j, A'_j, C'_j, T'_j) \neq (N_i, A_i, C_i, T_i)$
- $\text{Adv}_{\Pi}^{\text{mrae}}(\mathcal{A})$: the MRAE advantage of \mathcal{A} against Π
- DAEADs achieve MRAE security naturally!

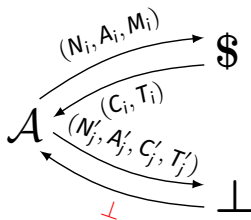
Security Notions: A Quick Recap

Misuse-resistant AE (MRAE)

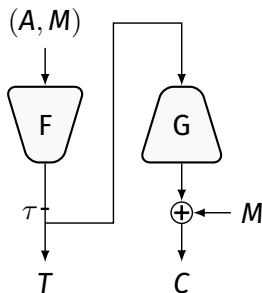
RW



IW

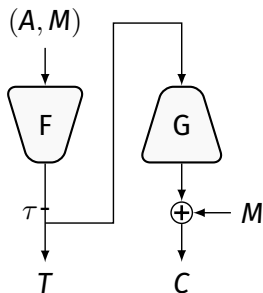


- Ideal world: a uniform random function $\$$ and the *reject* oracle \perp
- \mathcal{A} 's queries must satisfy $(N'_j, A'_j, C'_j, T'_j) \neq (N_i, A_i, C_i, T_i)$
- $\text{Adv}_{\Pi}^{\text{mrae}}(\mathcal{A})$: the MRAE advantage of \mathcal{A} against Π
- DAEADs achieve MRAE security naturally!



- Two main components:
 - F : a PRF
 - G : a random IV-based PRF
- Inverse-free
- Parallelizable
- Composition Bound [RS: EC '06, IM: ToSC '16]:

$$\mathbf{Adv}_{\text{SIV}}^{\text{mrae}}(\mathcal{A}) \leq \mathbf{Adv}_{\text{F}}^{\text{prf}}(\mathcal{B}) + \mathbf{Adv}_{\text{G}}^{\text{\$-prf}}(\mathcal{C}) + \frac{q}{2^\tau}$$



- Two main components:
 - F : a PRF
 - G : a random IV-based PRF
- Inverse-free
- Parallelizable
- Composition Bound [RS: EC '06, IM: ToSC '16]:

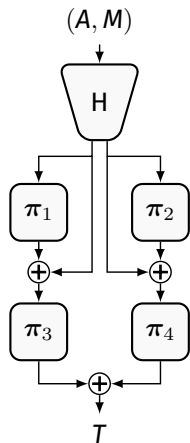
$$\mathbf{Adv}_{\text{SIV}}^{\text{mrae}}(\mathcal{A}) \leq \mathbf{Adv}_{\text{F}}^{\text{prf}}(\mathcal{B}) + \mathbf{Adv}_{\text{G}}^{\text{\$-prf}}(\mathcal{C}) + \frac{q}{2^\tau}$$

TODOs:

1. A BBB secure PRF component with $\tau > n$ bits of output
2. A BBB secure random IV-based PRF component

The PRF Component

Revisiting HtmB-p2 [CJN: AC '20]



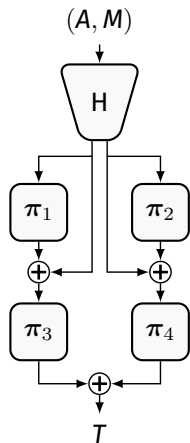
- Hashing solves two purposes:
 - Handling arbitrary length inputs
 - Inputs to $\pi_{\{1,2\}}$ have *controlled collisions*
 \implies Optimal Security for HtmB
- HtmB-p2 PRF Bound [CJN: AC '20, CDNPS EC '23]:

$$\text{Adv}_{\text{HtmB-p2}}^{\text{prf}}(\mathcal{A}) = O\left(\frac{q}{2^n} + q^2 \epsilon_{\text{coll}}\right)$$

- **Limitation:** only n -bit outputs

The PRF Component

Revisiting HtmB-p2 [CJN: AC '20]



- Hashing solves two purposes:
 - Handling arbitrary length inputs
 - Inputs to $\pi_{\{1,2\}}$ have *controlled collisions*
 \Rightarrow Optimal Security for HtmB

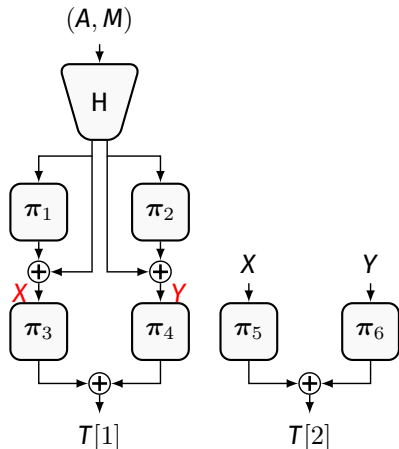
- HtmB-p2 PRF Bound [CJN: AC '20, CDNPS EC '23]:

$$\text{Adv}_{\text{HtmB-p2}}^{\text{prf}}(\mathcal{A}) = O\left(\frac{q}{2^n} + q^2 \epsilon_{\text{coll}}\right)$$

- **Limitation:** only n -bit outputs

The PRF Component

F^* : A BBB secure PRF with $2n$ -bit outputs



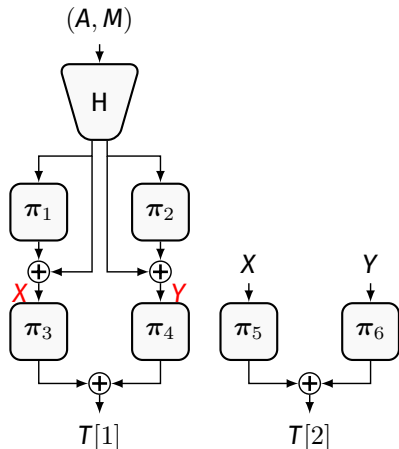
- HtmB-p2*:
 - Duplicates the HtmB-p2 finalization
 - Additional n bits at the cost of two calls
- F^* : HtmB-p2* with a PMAC+ like hash

F^* is optimally secure [for lengths up to $\sqrt{2^n}$]

$$\text{Adv}_{F^*}^{\text{prf}}(\mathcal{A}) = O\left(\frac{\sigma}{2^n}\right)$$

The PRF Component

F^* : A BBB secure PRF with $2n$ -bit outputs



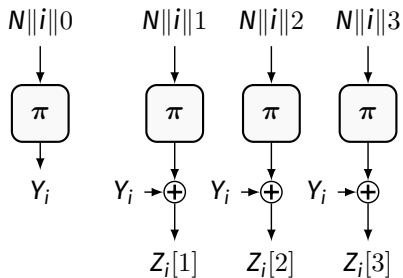
- HtmB-p2*:
 - Duplicates the HtmB-p2 finalization
 - Additional n bits at the cost of two calls
- F^* : HtmB-p2* with a PMAC+ like hash

F^* is optimally secure [for lengths up to $\sqrt{2^n}$]

$$\text{Adv}_{F^*}^{\text{prf}}(\mathcal{A}) = O\left(\frac{\sigma}{2^n}\right)$$

The Random-IV PRF Component (Option 1)

Revisiting CENC [Iwata: FSE '06]

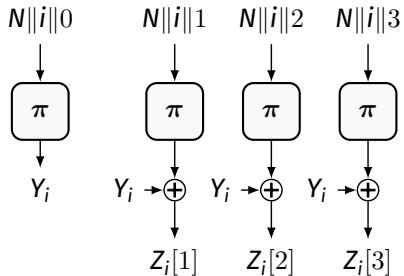


The i -th chunk of keystream ($r=3$)

- Keystream is generated in chunks of r blocks
- Fully parallelizable
- Rate $\approx \left(\frac{r}{r+1}\right)$
- Optimally secure if IVs are *unique* [IMV: ePrint '16]
- Limitations:
 - $|N| < n$ (we require $\approx 2n$)
 - Only *birthday-bound* \mathcal{S} -PRF secure

The Random-IV PRF Component (Option 1)

Revisiting CENC [Iwata: FSE '06]

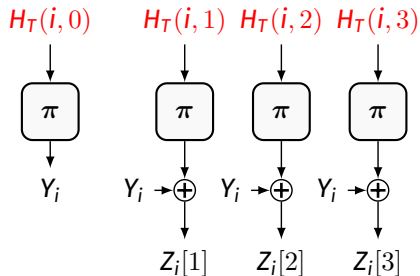


The i -th chunk of keystream ($r=3$)

- Keystream is generated in chunks of r blocks
- Fully parallelizable
- Rate $\approx \left(\frac{r}{r+1}\right)$
- Optimally secure if IVs are *unique* [IMV: ePrint '16]
- **Limitations:**
 - $|N| < n$ (we require $\approx 2n$)
 - Only *birthday-bound* $\$$ -PRF secure

The Random-IV PRF Component (Option 1)

GiantStar: A BBB secure random IV-based PRF



The i -th chunk of keystream ($r=3$)

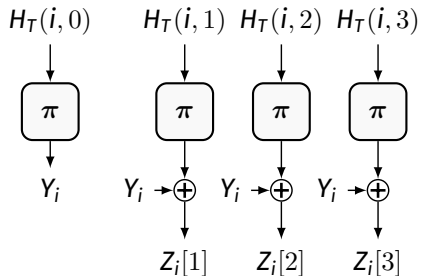
- CTR-based encoding \rightarrow *lightweight* hash
 - Use the random IV as key
- Inherits all the efficiency traits of CENC
- Secure if hash is 2-wise independent

GiantStar is BBB secure [for *moderately* large ℓ]

$$\text{Adv}_{\text{GiantStar}}^{\text{\$-prf}}(\mathcal{A}) = O\left(\frac{r\sigma}{2^n} + \frac{r\sigma^2\ell}{2^{2n}}\right)$$

The Random-IV PRF Component (Option 1)

GiantStar: A BBB secure random IV-based PRF



The i -th chunk of keystream ($r=3$)

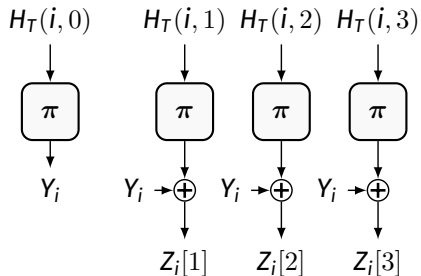
- CTR-based encoding \rightarrow *lightweight* hash
 - Use the random IV as key
- Inherits all the efficiency traits of CENC
- Secure if hash is 2-wise independent

GiantStar is BBB secure [for *moderately* large ℓ]

$$\text{Adv}_{\text{GiantStar}}^{\$-\text{prf}}(\mathcal{A}) = O\left(\frac{r\sigma}{2^n} + \frac{r\sigma^2\ell}{2^{2n}}\right)$$

The Random-IV PRF Component (Option 1)

GiantStar: A BBB secure random IV-based PRF



The i -th chunk of keystream ($r=3$)

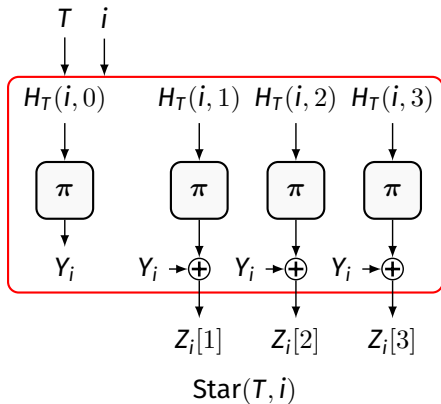
- CTR-based encoding \rightarrow *lightweight* hash
 - Use the random IV as key
- Inherits all the efficiency traits of CENC
- Secure if hash is 2-wise independent

GiantStar is BBB secure [for *moderately* large ℓ]

$$\text{Adv}_{\text{GiantStar}}^{\$-\text{prf}}(\mathcal{A}) = O\left(\frac{r\sigma}{2^n} + \frac{r\sigma^2 \ell}{2^{2n}}\right)$$

The Random-IV PRF Component (Option 2)

Star: A fixed-length BBB secure random IV-based PRF



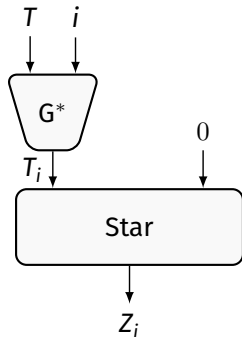
- $\text{Star} \equiv \text{GiantStar}$ with
 - Fixed chunk index i
 - Restricted to $\leq r$ -block outputs

Star is optimally secure

$$\text{Adv}_{\text{Star}}^{\text{\$-prf}}(\mathcal{A}) = o\left(\frac{rq}{2^n}\right)$$

The Random-IV PRF Component (Option 2)

Snowflake: A length-independent BBB secure random IV-based PRF



The i -th chunk of keystream

- Fresh $2n$ -bit randomness per chunk

$$\mathbf{Adv}_{\text{Snowflake}}^{\$-\text{prf}}(\mathcal{A}) \leq \mathbf{Adv}_{\text{Star}}^{\$-\text{prf}}(\mathcal{B}) + \mathbf{Adv}_{G^*}^{\$-\text{prf}}(\mathcal{C})$$

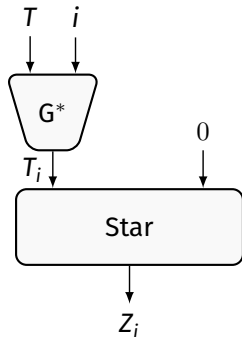
- G^* must have *length-independent* bound!
- G^* can be relatively heavier
 - in the paper: 6 calls per chunk

Snowflake is optimally secure

$$\mathbf{Adv}_{\text{Snowflake}}^{\$-\text{prf}}(\mathcal{A}) = O\left(\frac{r\sigma}{2^n}\right)$$

The Random-IV PRF Component (Option 2)

Snowflake: A length-independent BBB secure random IV-based PRF



The i -th chunk of keystream

- Fresh $2n$ -bit randomness per chunk

$$\mathbf{Adv}_{\text{Snowflake}}^{\$-\text{prf}}(\mathcal{A}) \leq \mathbf{Adv}_{\text{Star}}^{\$-\text{prf}}(\mathcal{B}) + \mathbf{Adv}_{G^*}^{\$-\text{prf}}(\mathcal{C})$$

- G^* must have *length-independent* bound!
- G^* can be relatively heavier
 - in the paper: 6 calls per chunk

Snowflake is optimally secure

$$\mathbf{Adv}_{\text{Snowflake}}^{\$-\text{prf}}(\mathcal{A}) = O\left(\frac{r\sigma}{2^n}\right)$$

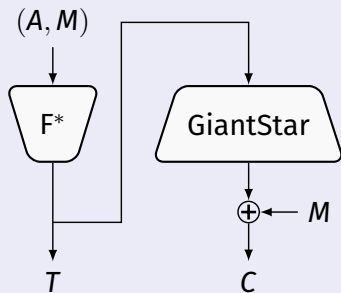
Our Contributions

Two misuse-resistant BBB-secure AEAD modes

Our Contributions

Two misuse-resistant BBB-secure AEAD modes

DENC1



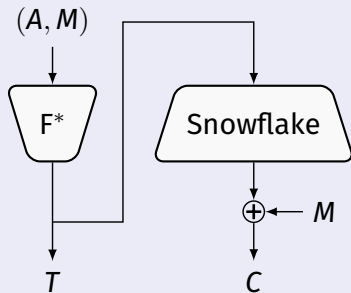
- Highly *parallelizable*
- Tag size $\tau = 2n$ -bit
- Max. input length $\ell \leq \sqrt{2^n}$ -block
- Rate $\geq \left(\frac{r}{2r+0.5}\right)$ (≈ 0.498 for $r = 64$)
- BBB secure for *moderate* message lengths

$$\mathbf{Adv}_{\text{DENC1}}^{\text{mrae}}(\mathcal{A}) = O\left(\frac{r\sigma^2\ell}{2^{2n}}\right)$$

Our Contributions

Two misuse-resistant BBB-secure AEAD modes

DENC2



- Highly parallelizable
- Tag size $\tau = 2n$ -bit
- Max. input length $\ell \leq \sqrt{2^n}$ -block
- Rate $\geq \left(\frac{r}{2r+3.5}\right)$ (≈ 0.486 for $r = 64$)
- **Length-independent** optimal security

$$\text{Adv}_{\text{DENC1}}^{\text{mrae}}(\mathcal{A}) = O\left(\frac{r\sigma}{2^n}\right)$$

Making GCM Great Again: Toward Full Security and Longer Nonces

Woohyuk Chung¹ Seongha Hwang¹ Seongkwang Kim²

Byeonghak Lee² Jooyoung Lee¹

¹KAIST, Korea

²Samsung SDS, Korea

Eurocrypt 2025

2025. 05. 06.

Same Motivation, Different Goal

Recall: We require BBB-secure AEAD with low nonce misusing risk.

1. Design a misuse-resistant AE

- AES-GCM-SIV, DENC1, DENC2, ...
- Best for security, but **inherently two pass**

2. Design a nonce-based AE with extended nonces

- DNDK-GCM: requires carefully generated nonces and BC with $2n$ -bit key

Same Motivation, Different Goal

Recall: We require BBB-secure AEAD with low nonce misusing risk.

1. Design a misuse-resistant AE
 - AES-GCM-SIV, DENC1, DENC2, ...
 - Best for security, but **inherently two pass**
2. Design a nonce-based AE with extended nonces
 - DNDK-GCM: requires carefully generated nonces and BC with $2n$ -bit key

Same Motivation, Different Goal

Recall: We require BBB-secure AEAD with low nonce misusing risk.

1. Design a misuse-resistant AE
 - AES-GCM-SIV, DENC1, DENC2, ...
 - Best for security, but **inherently two pass**
2. Design a nonce-based AE with extended nonces
 - DNDK-GCM: requires carefully generated nonces and BC with $2n$ -bit key

Our Goal:

- Block cipher based AE with **full security**
+ Provably secure under standard PRP assumption
- Efficiency is comparable to GCM
- Support **extended nonces** or provide **nonce misuse resistance**
- Support arbitrary length message

Starting Point: CENC

Cipher-based ENCrption (CENC)

- CTR-type encryption mode with full security

$$\text{Adv}_{\text{CENC}[E,r]}^{\text{prf}}(q, \sigma, l) \leq O\left(\frac{\sigma}{2^n}\right)$$

- limitation: $|\text{nonce}| + |\text{counter}| \leq n$
 \Rightarrow still have nonce misusing risk and short length limitation

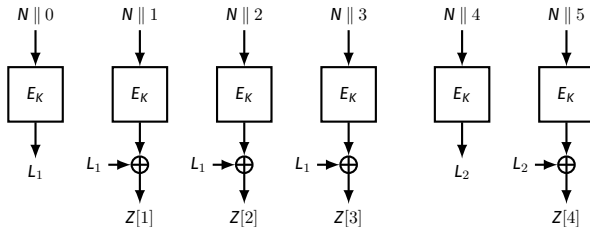


Figure: The first 4 keystream blocks from $\text{CENC}[E_K, w](N, \cdot)$ with $w = 3$.

Building Blocks - eCTR

enhanced CTR (eCTR) (\simeq GiantStar !)

- almost fully secure variable output length PRF (VOL-PRF) with $2n$ -bit random IV

$$\mathbf{Adv}_{\text{eCTR}[E,r]}^{\$-\text{prf}}(\mathcal{A}) \leq O\left(\frac{r\sigma}{2^n} + \frac{r\sigma^2 l}{2^{2n}}\right)$$

- limitation: requires **random IV**
 \Rightarrow enough for iv-based AE, but we want nonce-based

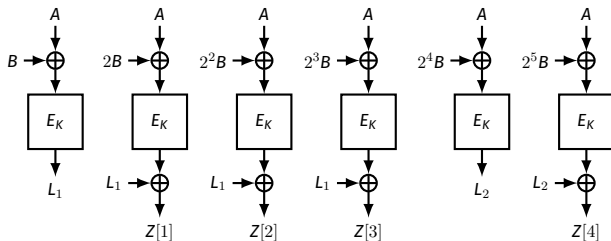


Figure: The first 4 blocks from $\text{eCTR}[E_K, w](A, B)$ with $w = 3$.

Building Blocks - HteC

Hash-then-eCTR (HteC)

- almost fully secure variable input/output length PRF (VIL-VOL-PRF)

$$\text{Adv}_{\text{HteC}[H,E,w]}^{\text{prf}}(\mathcal{A}) \leq O\left(\frac{w\sigma}{2^n} + \frac{w\sigma^2 l}{2^{2n}}\right)$$

where H is δ -universal hash (UH)

- UH-then-PRP outputs ($= A, B$) are not fully random but enough for eCTR input

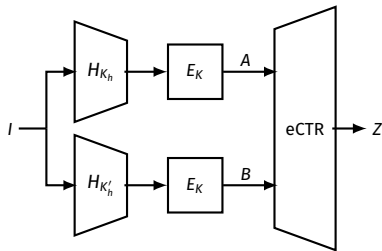
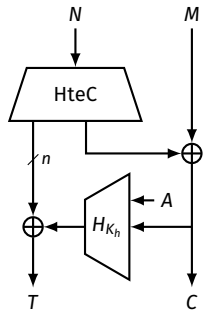


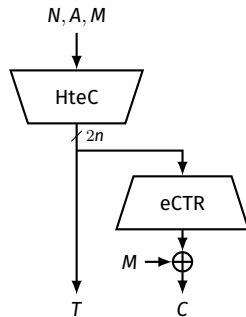
Figure: The HteC VIL-VOL pseudorandom function.

Our Contribution

eGCM/eGCM-SIV: enhanced variant of GCM/GCM-SIV



(a) Encryption of eGCM

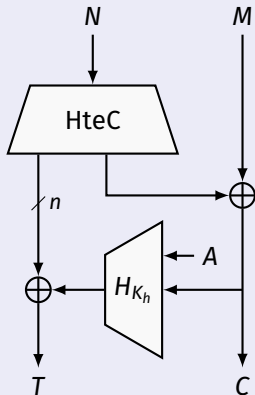


(b) Encryption of eGCM-SIV

Figure: The eGCM and eGCM-SIV AE schemes. A nonce, an associated data, and a message are denoted N , A and M , respectively

Our Contribution

eGCM

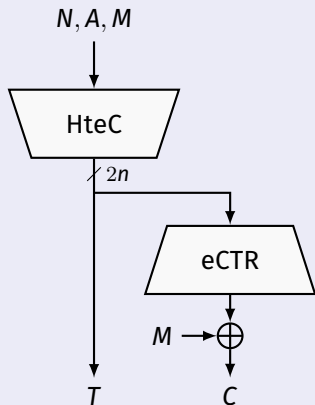


- Replace CTR to HteC
- Support **extended nonces**
- Support encrypting arbitrary length messages
- BBB secure for *moderate* message lengths

$$\text{Adv}_{\text{eGCM}}^{\text{nae}}(\mathcal{A}) = O\left(\frac{r\sigma^2\ell}{2^{2n}}\right)$$

Our Contribution

eGCM-SIV



- Use HteC as PRF and replace CTR to eCTR
- Support encrypting arbitrary length messages
- BBB secure for *moderate* message lengths

$$\text{Adv}_{\text{eGCM-SIV}}^{\text{dae}}(\mathcal{A}) = O\left(\frac{r\sigma^2\ell}{2^{2n}}\right)$$

Comparison

AEAD	Rate	Security	
		NR	NM
OCB3	1	$n/2$	-
GCM	$1/2$	$n/2$	-
CIP, CHM, mGCM, eGCM	$\lesssim 1/2^\dagger$	n	-
AES-GCM-SIV	$1/2$	n	$n/2$
SCM	$1/2$	n	$n/2$
CWC+	$\lesssim 1/2^\dagger$	$3n/4$	$n/2$ (auth only)
eGCM-SIV, DENC1, DENC2	$\lesssim 1/2^\dagger$	n	n

[†] Depends on the parameter w , while we write $\lesssim 1/2$ since the rate approaches $1/2$ as w increases and w can be set to a large enough value.

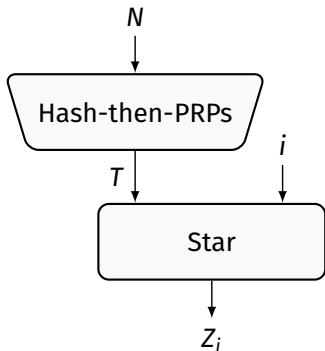
Table: Comparison of eGCM, eGCM-SIV, DENC1 and DENC2 and other block cipher based AE schemes. The maximum message length ($= l$) is assumed to be a small constant. Note that **DENC2** has length-independent security.

Benchmark

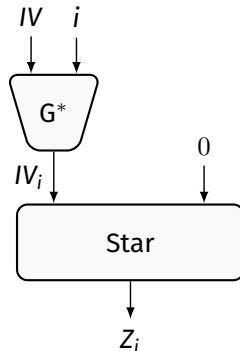
AEAD	Message		
	1KB	4KB	64KB
OCB3	0.52	0.47	0.45
GCM	1.65	1.02	0.83
eGCM	0.93	0.89	0.88
AES-GCM-SIV	1.33	1.07	0.99
SCM	1.19	1.11	1.07
eGCM-SIV	1.33	1.15	1.12
DENC1	1.31	1.20	1.18
DENC2	1.42	1.38	1.32

Table: Benchmark of eGCM, eGCM-SIV, DAE1 and DAE2 and other block cipher based AE schemes. Throughput is measured in cycles per byte, for empty associated data.

HteC vs SnowFlake

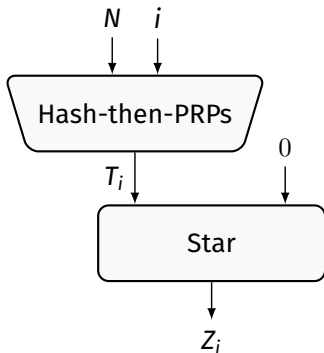


- Use arbitrary length **nonces**
- Simpler compressing function
- Length-dependent security



- Use random IVs (enough for SIV)
- **Length-independent** security
- G^* is heavy!

Combining Two Papers: HteC + SnowFlake



- G^* is replaced by Hash-then-PRPs \Rightarrow faster and support nonce!
- VIL-VOL-PRF with (output) **length-independent** security
- Can be used to construct fully secure NAE and DAE

Conclusion

Towards Optimally Secure DAEs

- DENC1: almost fully secure DAE
- DENC2: fully secure DAE (**length-independent** security)

Making GCM Great Again

- HteC: almost fully secure VIL-VOL-PRF
- eGCM: almost fully secure NAE with **extended nonces**
- eGCM-SIV: almost fully secure DAE

Our results can also be applied to:

- Accordion ciphers: Hash-CTR-Hash \Rightarrow Hash-(eCTR/SnowFlake)-Hash
- Nonce-key derivation: HteC and HteC+SnowFlake are PRF

Thank you for your attention!