CISPA
HELMHOLTZ CENTER FOR INFORMATION SECURITY

Inria

Automated Analysis of Protocols that use Authenticated Encryption:

Analysing the Impact of the Subtle Differences between AEADs on Protocol Security


Cas Cremers*


Alexander Dax*


Charlie Jacomme‡


Mang Zhao*

* *CISPA Helmholtz Center for Information Security*

‡ *INRIA Paris*

Some weird behaviour in protocols

Some weird behaviour in protocols

Facebook's Message Franking:

- Abuse reporting mechanism within E2EE communication
- [DGRW18] the reported message seen by the server is not the same as the one seen by the reporting user

Some weird behaviour in protocols

Facebook's Message Franking:

- Abuse reporting mechanism within E2EE communication
- [DGRW18] the reported message seen by the server is not the same as the one seen by the reporting user

SFrame:

- E2EE group communication protocol for audio/video
- [IIM21] impersonate other group members

Some weird behaviour in protocols

Facebook's Message Franking:

- Abuse reporting mechanism within E2EE communication
- [DGRW18] the reported message seen by the server is not the same as the one seen by the reporting user

SFrame:

- E2EE group communication protocol for audio/video
- [IIM21] impersonate other group members by collision

YubiHSM:

- Store cryptographic secrets and offer API for crypto operations
- [KS12] Full secret key leaked

Some weird behaviour in protocols

Facebook's Message Franking:

- Abuse reporting mechanism within E2EE communication
- [DGRW18] the reported message seen by the server is not the same as the one seen by the reporting user

SFrame:

- E2EE group communication protocol for audio/video
- [IIM21] impersonate other group members by collision

YubiHSM:

- Store cryptographic secrets and offer API for crypto operations
- [KS12] Full secret key leaked

Where do those attacks come from?

AEAD is complex!

CISPA
HELMHOLTZ CENTER FOR INFORMATION SECURITY

AEAD is complex!

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Authenticated Encryption with associated data (AEAD):

- symmetric encryption
- data is integrity protected

AEAD is complex!

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Authenticated Encryption with associated data (AEAD):

- symmetric encryption
- data is integrity protected
- Often 4 inputs

AEAD is complex!

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Authenticated Encryption with associated data (AEAD):

- symmetric encryption
- data is integrity protected
- Often 4 inputs
  - Secret key
  - data/message

AEAD is complex!

Authenticated Encryption with associated data (AEAD):

- symmetric encryption
- data is integrity protected
- Often 4 inputs
  - Secret key
  - data/message
  - associated data (e.g., meta info)
    - In the clear,
      but also integrity protected

AEAD is complex!

Authenticated Encryption with associated data (AEAD):

- symmetric encryption
- data is integrity protected
- Often 4 inputs
  - Secret key
  - data/message
  - associated data (e.g., meta info)
    - In the clear,
      but also integrity protected
  - Entropy source:
    - Nonces
    - Counters

AEAD is complex!

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Authenticated Encryption with associated data (AEAD):

- symmetric encryption
- data is integrity protected
- Often 4 inputs
  - Secret key
  - data/message
  - associated data (e.g., meta info)
    - In the clear,
      but also integrity protected
  - Entropy source:
    - Nonces
    - Counters

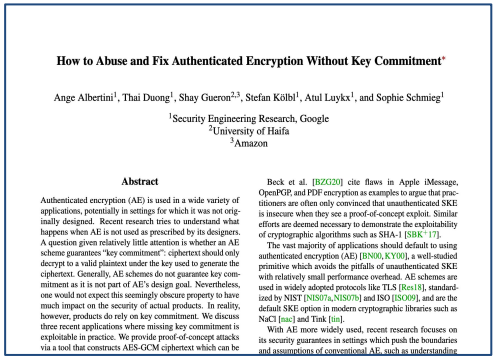
Many competing definitions as we speak!

AEAD is complex!

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Authenticated Encryption with associated data (AEAD):

- symmetric encryption
- data is integrity protected
- Often 4 inputs
  - Secret key
  - data/message
  - associated data (e.g., meta info)
    - In the clear,
      but also integrity protected
  - Entropy source:
    - Nonces
    - Counters



How to Abuse and Fix Authenticated Encryption Without Key Commitment*

Ange Albertini[1], Thai Duong[1], Shay Gueron[2,3], Stefan Kölbl[1], Atul Luykx[1], and Sophie Schmieg[1]

[1]Security Engineering Research, Google
[2]University of Haifa
[3]Amazon

Abstract

Authenticated encryption (AE) is used in a wide variety of applications, potentially in settings for which it was not originally designed. Recent research tries to understand what happens when AE is not used as prescribed by its designers. A question given relatively little attention is whether an AE scheme guarantees "key commitment": ciphertext should only decrypt to a valid plaintext under the key used to generate the ciphertext. Generally, AE schemes do not guarantee key commitment as it is not part of AE's design goal. Nevertheless, one would not expect this seemingly obscure property to have much impact on the security of actual products. In reality, however, products do rely on key commitment. We discuss three recent applications where missing key commitment is exploitable in practice. We provide proof-of-concept attacks via a tool that constructs AES-GCM ciphertext which can be

Beck et al. [BZG20] cite flaws in Apple iMessage, OpenPGP, and PDF encryption as examples to argue that practitioners are often only convinced that unauthenticated SKE is insecure when they see a proof-of-concept exploit. Similar efforts are deemed necessary to demonstrate the exploitability of cryptographic algorithms such as SHA-1 [SBK+17].

The vast majority of applications should default to using authenticated encryption (AE) [BN00, KY00], a well-studied primitive which avoids the pitfalls of unauthenticated SKE with relatively small performance overhead. AE schemes are used in widely adopted protocols like TLS [Res18], standardized by NIST [NIST07a, NIST07b] and ISO [ISO09], and are the default SKE option in modern cryptographic libraries such as NaCl [nac] and Tink [tin].

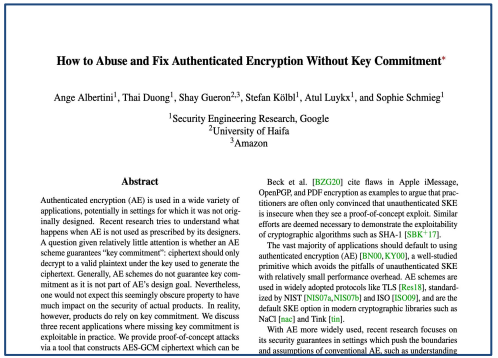
With AE more widely used, recent research focuses on its security guarantees in settings which push the boundaries and assumptions of conventional AE, such as understanding

Many competing definitions as we speak!

AEAD is complex!

Authenticated Encryption with associated data (AEAD):

- symmetric encryption
- data is integrity protected
- Often 4 inputs
  - Secret key
  - data/message
  - associated data (e.g., meta info)
    - In the clear,
      but also integrity protected
  - Entropy source:
    - Nonces
    - Counters



How to Abuse and Fix Authenticated Encryption Without Key Commitment*

Ange Albertini[1], Thai Duong[1], Shay Gueron[2,3], Stefan Kölbl[1], Atul Luykx[1], and Sophie Schmieg[1]

[1]Security Engineering Research, Google
[2]University of Haifa
[3]Amazon

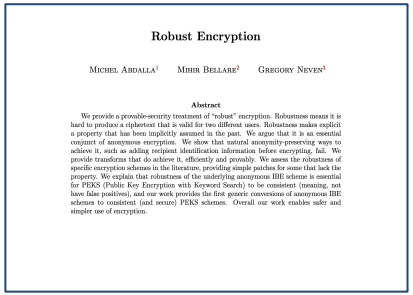
Abstract

Authenticated encryption (AE) is used in a wide variety of applications, potentially in settings for which it was not originally designed. Recent research tries to understand what happens when AE is not used as prescribed by its designers. A question given relatively little attention is whether an AE scheme guarantees "key commitment": ciphertext should only decrypt to a valid plaintext under the key used to generate the ciphertext. Generally, AE schemes do not guarantee key commitment as it is not part of AE's design goal. Nevertheless, one would not expect this seemingly obscure property to have much impact on the security of actual products. In reality, however, products do rely on key commitment. We discuss three recent applications where missing key commitment is exploitable in practice. We provide proof-of-concept attacks via a tool that constructs AES-GCM ciphertext which can be

Beck et al. [BZG20] cite flaws in Apple iMessage, OpenPGP, and PDF encryption as examples to argue that practitioners are often only convinced that unauthenticated SKE is insecure when they see a proof-of-concept exploit. Similar efforts are deemed necessary to demonstrate the exploitability of cryptographic algorithms such as SHA-1 [SBK+17].

The vast majority of applications should default to using authenticated encryption (AE) [BN00, KY00], a well-studied primitive which avoids the pitfalls of unauthenticated SKE with relatively small performance overhead. AE schemes are used in widely adopted protocols like TLS [Res18], standardized by NIST [NIST07a, NIST07b] and ISO [ISO09], and are the default SKE option in modern cryptographic libraries such as NaCl [nac] and Tink [tin].

With AE more widely used, recent research focuses on its security guarantees in settings which push the boundaries and assumptions of conventional AE, such as understanding



Robust Encryption

Michel Abdalla[1] Mihir Bellare[2] Gregory Neven[3]

Abstract

We provide a provable-security treatment of "robust" encryption. Robustness means it is hard to produce a ciphertext that is valid for two different users. Robustness makes explicit a property that has been implicitly assumed in the past. We argue that it is an essential conjunct of anonymous encryption. We show that natural anonymity-preserving ways to achieve it, such as adding recipient identification information before encrypting, fail. We provide transforms that do achieve it, efficiently and provably. We assess the robustness of specific encryption schemes in the literature, providing simple patches for some that lack the property. We explain that robustness of the underlying anonymous IBE scheme is essential for PEKS (Public Key Encryption with Keyword Search) to be consistent (meaning, not have false positives), and our work provides the first generic conversions of anonymous IBE schemes to consistent [and secure] PEKS schemes. Overall our work enables safer and simpler use of encryption.

Many competing definitions as we speak!

AEAD is complex!

Authenticated Encryption with associated data (AEAD):

- symmetric encryption
- data is integrity protected
- Often 4 inputs
  - Secret key
  - data/message
  - associated data (e.g., meta info)
    - In the clear,
      but also integrity protected
  - Entropy source:
    - Nonces
    - Counters

How to Abuse and Fix Authenticated Encryption Without Key Commitment*

Ange Albertini[1], Thai Duong[1], Shay Gueron[2,3], Stefan Kölbl[1], Atul Luykx[1], and Sophie Schmieg[1]

[1]Security Engineering Research, Google
[2]University of Haifa
[3]Amazon

Abstract

Authenticated encryption (AE) is used in a wide variety of applications, potentially in settings for which it was not originally designed. Recent research tries to understand what happens when AE is not used as prescribed by its designers.

Beck et al. [BZG20] cite flaws in Apple iMessage, OpenPGP, and PDF encryption as examples to argue that practitioners are often only convinced that unauthenticated SKE is insecure when they see a proof-of-concept exploit. Similar efforts are deemed necessary to demonstrate the exploitability

Partitioning Oracle Attacks

Julia Len Paul Grubbs Thomas Ristenpart

Cornell Tech

Abstract

In this paper we introduce *partitioning oracles*, a new class of decryption error oracles which, conceptually, take a ciphertext as input and output whether the decryption key belongs to some known subset of keys. Partitioning oracles can arise when encryption schemes are not committing with respect to their keys. We detail adaptive chosen ciphertext attacks that exploit partitioning oracles to efficiently recover passwords and de-anonymize anonymous communications. The attacks utilize efficient key multi-collision algorithms — a cryptanalytic goal that we define — against widely used authenticated encryption with associated data (AEAD) schemes, including AES-GCM, XSalsa20/Poly1305, and ChaCha20/Poly1305.

We build a practical partitioning oracle attack that quickly recovers passwords from Shadowsocks proxy servers. We

for encrypted messaging [21, 29].

We introduce partitioning oracle attacks, a new type of CCA. Briefly, a partitioning oracle arises when an adversary can: (1) efficiently craft ciphertexts that successfully decrypt under a large number of potential keys, and (2) can submit such ciphertexts to a system that reveals whether decryption under a target secret key succeeds. This enables an attacker to learn information about the secret key. The main cryptanalytic step for our attacks is constructing (what we call) key multi-collisions, in which a single AEAD ciphertext can be built such that decryption succeeds under some number k of keys. We formalize this cryptanalytic goal and give an algorithm for computing key multi-collisions for AES-GCM. It builds key multi-collision ciphertexts of length $O(k)$ in $O(k^2)$ time, making them reasonably scalable even to large k. We

Robust Encry

Michel Abdalla[1] Mihir Bella

Abstract

We provide a provable-security treatment of "robu
had to produce a ciphertext that is valid for two dif
a property that has been implicitly assumed in the
conjunct of anonymous encryption. We show that
achieve it, such as adding recipient identification in
provide transforms that do achieve it, efficiently and
specific encryption schemes in the literature, providin
property. We explain that robustness of the underly
for PEKS (Public Key Encryption with Keyword S
have false positives), and our work provides the first
schemes to consistent [and secure] PEKS schemes.
simpler use of encryption.

Many competing definitions as we speak!

AEAD is complex!

Authenticated Encryption with associated data (AEAD):

- symmetric encryption
- data is integrity protected
- Often 4 inputs
  - Secret key
  - data/message
  - associated data (e.g., meta info)
    - In the clear,
      but also integrity protected
  - Entropy source:
    - Nonces
    - Counters

Many competing definitions as we speak!

AEAD is complex!

Authenticated Encryption with associated data (AEAD):

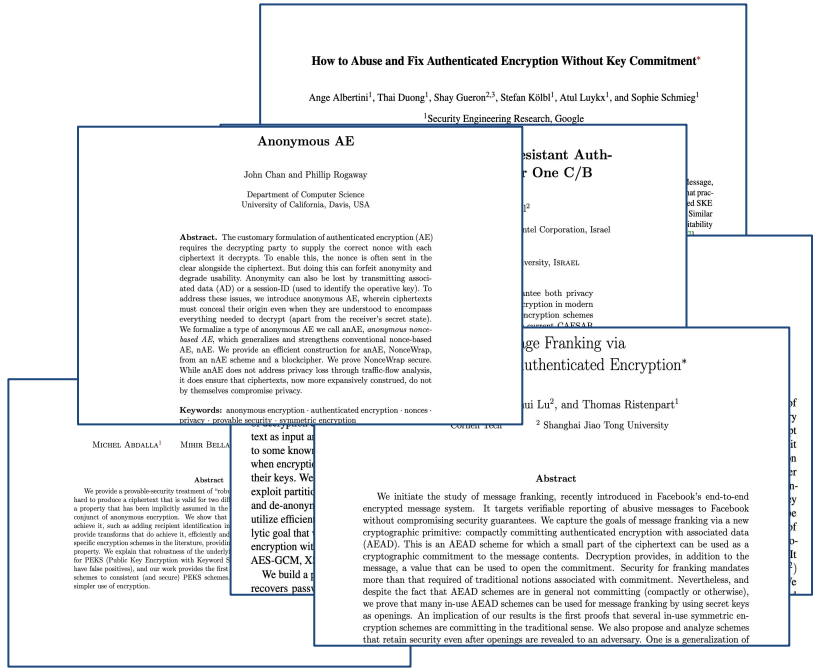
- symmetric encryption
- data is integrity protected
- Often 4 inputs
  - Secret key
  - data/message
  - associated data (e.g., meta info)
    - In the clear,
      but also integrity protected
  - Entropy source:
    - Nonces
    - Counters

Many competing definitions as we speak!

AEAD is complex!

Authenticated Encryption with associated data (AEAD):

- symmetric encryption
- data is integrity protected
- Often 4 inputs
  - Secret key
  - data/message
  - associated data (e.g., meta info)
    - In the clear,
      but also integrity protected
  - Entropy source:
    - Nonces
    - Counters

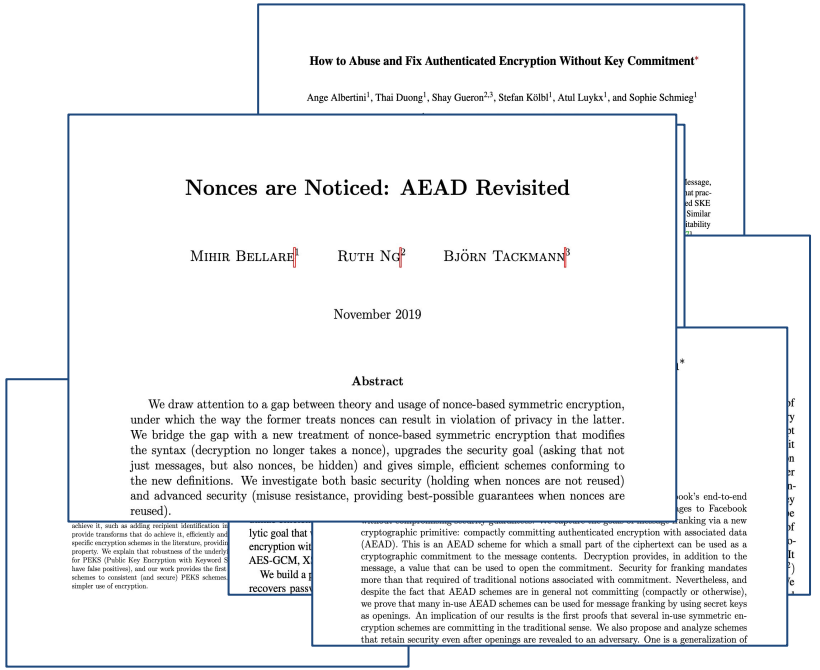


Many competing definitions as we speak!

AEAD is complex!

Authenticated Encryption with associated data (AEAD):

- symmetric encryption
- data is integrity protected
- Often 4 inputs
  - Secret key
  - data/message
  - associated data (e.g., meta info)
    - In the clear,
      but also integrity protected
  - Entropy source:
    - Nonces
    - Counters

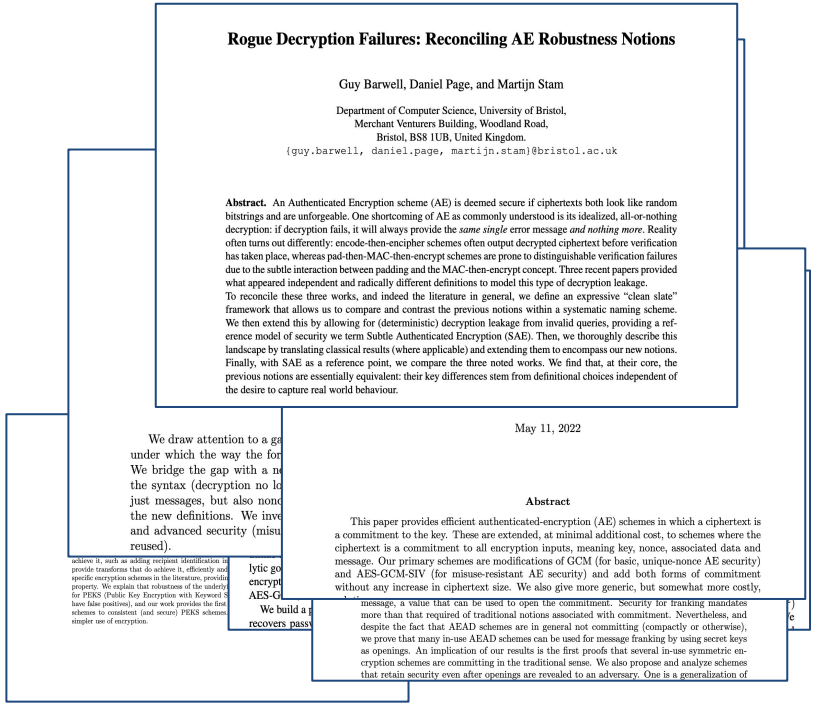


Many competing definitions as we speak!

AEAD is complex!

Authenticated Encryption with associated data (AEAD):

- symmetric encryption
- data is integrity protected
- Often 4 inputs
  - Secret key
  - data/message
  - associated data (e.g., meta info)
    - In the clear,
      but also integrity protected
  - Entropy source:
    - Nonces
    - Counters



Rogue Decryption Failures: Reconciling AE Robustness Notions

Guy Barwell, Daniel Page, and Martijn Stam

Department of Computer Science, University of Bristol,
Merchant Venturers Building, Woodland Road,
Bristol, BS8 1UB, United Kingdom.
{guy.barwell, daniel.page, martijn.stam}@bristol.ac.uk

Abstract. An Authenticated Encryption scheme (AE) is deemed secure if ciphertexts both look like random bitstrings and are unforgeable. One shortcoming of AE as commonly understood is its idealized, all-or-nothing decryption: if decryption fails, it will always provide the *same single* error message *and nothing more*. Reality often turns out differently: encode-then-encipher schemes often output decrypted ciphertext before verification has taken place, whereas pad-then-MAC-then-encrypt schemes are prone to distinguishable verification failures due to the subtle interaction between padding and the MAC-then-encrypt concept. Three recent papers provided what appeared independent and radically different definitions to model this type of decryption leakage.
To reconcile these three works, and indeed the literature in general, we define an expressive "clean slate" framework that allows us to compare and contrast the previous notions within a systematic naming scheme. We then extend this by allowing for (deterministic) decryption leakage from invalid queries, providing a reference model of security we term Subtle Authenticated Encryption (SAE). Then, we thoroughly describe this landscape by translating classical results (where applicable) and extending them to encompass our new notions. Finally, with SAE as a reference point, we compare the three noted works. We find that, at their core, the previous notions are essentially equivalent: their key differences stem from definitional choices independent of the desire to capture real world behaviour.

May 11, 2022

Abstract

This paper provides efficient authenticated-encryption (AE) schemes in which a ciphertext is a commitment to the key. These are extended, at minimal additional cost, to schemes where the ciphertext is a commitment to all encryption inputs, meaning key, nonce, associated data and message. Our primary schemes are modifications of GCM (for basic, unique-nonce AE security) and AES-GCM-SIV (for misuse-resistant AE security) and add both forms of commitment without any increase in ciphertext size. We also give more generic, but somewhat more costly,

message, a value that can be used to open the commitment. Security for franking mandates more than that required of traditional notions associated with commitment. Nevertheless, and despite the fact that AEAD schemes are in general not committing (compactly or otherwise), we prove that many in-use AEAD schemes can be used for message franking by using secret keys as openings. An implication of our results is the first proofs that several in-use symmetric encryption schemes are committing in the traditional sense. We also propose and analyze schemes that retain security even after openings are revealed to an adversary. One is a generalization of

Many competing definitions as we speak!

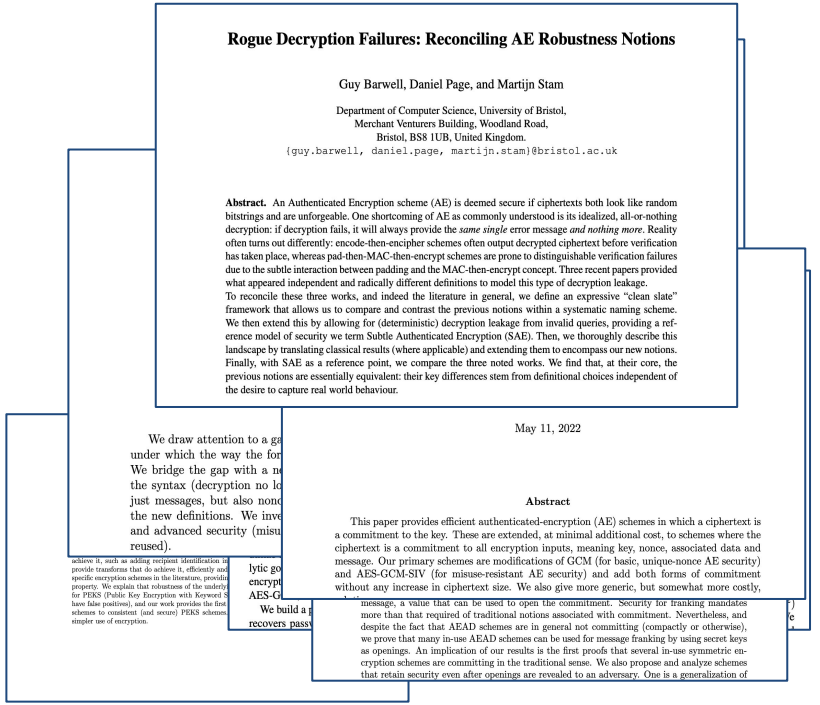
AEAD is complex!

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Authenticated Encryption with associated data (AEAD):

- symmetric encryption
- data is integrity protected
- Often 4 inputs
  - Secret key
  - data/message
  - associated data (e.g., meta info)
    - In the clear,
      but also integrity protected
  - Entropy source:
    - Nonces
    - Counters

→ What guarantees does it give protocol designers?



Rogue Decryption Failures: Reconciling AE Robustness Notions

Guy Barwell, Daniel Page, and Martijn Stam

Department of Computer Science, University of Bristol,
Merchant Venturers Building, Woodland Road,
Bristol, BS8 1UB, United Kingdom.
{guy.barwell, daniel.page, martijn.stam}@bristol.ac.uk

Abstract. An Authenticated Encryption scheme (AE) is deemed secure if ciphertexts both look like random bitstrings and are unforgeable. One shortcoming of AE as commonly understood is its idealized, all-or-nothing decryption: if decryption fails, it will always provide the *same single* error message *and nothing more*. Reality often turns out differently: encode-then-encipher schemes often output decrypted ciphertext before verification has taken place, whereas pad-then-MAC-then-encrypt schemes are prone to distinguishable verification failures due to the subtle interaction between padding and the MAC-then-encrypt concept. Three recent papers provided what appeared independent and radically different definitions to model this type of decryption leakage.
To reconcile these three works, and indeed the literature in general, we define an expressive "clean slate" framework that allows us to compare and contrast the previous notions within a systematic naming scheme. We then extend this by allowing for (deterministic) decryption leakage from invalid queries, providing a reference model of security we term Subtle Authenticated Encryption (SAE). Then, we thoroughly describe this landscape by translating classical results (where applicable) and extending them to encompass our new notions. Finally, with SAE as a reference point, we compare the three noted works. We find that, at their core, the previous notions are essentially equivalent: their key differences stem from definitional choices independent of the desire to capture real world behaviour.

May 11, 2022

Abstract

This paper provides efficient authenticated-encryption (AE) schemes in which a ciphertext is a commitment to the key. These are extended, at minimal additional cost, to schemes where the ciphertext is a commitment to all encryption inputs, meaning key, nonce, associated data and message. Our primary schemes are modifications of GCM (for basic, unique-nonce AE security) and AES-GCM-SIV (for misuse-resistant AE security) and add both forms of commitment without any increase in ciphertext size. We also give more generic, but somewhat more costly,

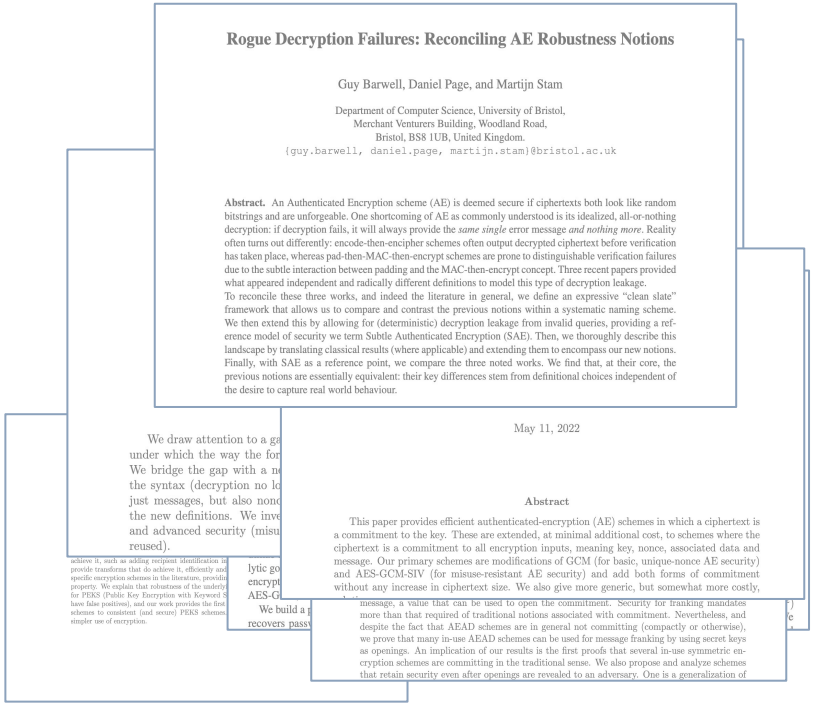
message, a value that can be used to open the commitment. Security for franking mandates more than that required of traditional notions associated with commitment. Nevertheless, and despite the fact that AEAD schemes are in general not committing (compactly or otherwise), we prove that many in-use AEAD schemes can be used for message franking by using secret keys as openings. An implication of our results is the first proofs that several in-use symmetric encryption schemes are committing in the traditional sense. We also propose and analyze schemes that retain security even after openings are revealed to an adversary. One is a generalization of

Many competing definitions as we speak!

AEAD is complex!

Authenticated Encryption with associated data (AEAD):

- symmetric encryption
- data is integrity protected
- Often 4 inputs
  - Secret key
  - data/message
  - associated data (e.g., meta info)
    - In the clear, but also integrity protected
  - Entropy source:
    - Nonces
    - Counters



→ What guarantees does it give protocol designers?

tions as we speak!

Many ways to misuse and misunderstand AEADs

Many different attack surfaces from the AEAD

Facebook's Message Franking

SFrame

YubiHSM

Many different attack surfaces from the AEAD

Facebook's Message Franking

→ [DGRW18] A single encrypted message can be decrypted under 2 distinct keys to 2 different meaningful messages

SFrame

YubiHSM

Many different attack surfaces from the AEAD

Facebook's Message Franking

→ [DGRW18] A single encrypted message can be decrypted under 2 distinct keys to 2 different meaningful messages

SFrame

→ [IIM21] instead of signing message, only signed authentication tag which makes it possible to impersonate peers

YubiHSM

Many different attack surfaces from the AEAD

Facebook's Message Franking

→ [DGRW18] A single encrypted message can be decrypted under 2 distinct keys to 2 different meaningful messages

SFrame

→ [IIM21] instead of signing message, only signed authentication tag which makes it possible to impersonate peers

YubiHSM

→ [KS12] allowing to reuse nonces leads leakage of secret key

Many different attack surfaces from the AEAD

Facebook's Message Franking

→ [DGRW18] A single encrypted message can be decrypted under 2 distinct keys to 2 different meaningful messages

SFrame

→ [IIM21] instead of signing message, only signed authentication tag which makes it possible to impersonate peers

YubiHSM

→ [KS12] allowing to reuse nonces leads leakage of secret key

How can we prevent such attacks?

Security analysis

Security analysis, but it is hard…

Security analysis, but it is hard…

Two major models: Computational and Symbolic

Security analysis, but it is hard…

Two major models: Computational and Symbolic

Computational Model:

- Does not scale well
- Limited automation

Symbolic Model:

- Authenticated encryption modelled very coarsely
- Traditional representation misses attacks

Security analysis, but it is hard…

Two major models: Computational and Symbolic

Computational Model:

- Does not scale well
- Limited automation

Symbolic Model:

- Authenticated encryption modelled very coarsely
- Traditional representation misses attacks

Attack finding until now was manual effort. Can we do better?

Security analysis, but it is hard…

Two major models: Computational and Symbolic

Computational Model:

- Does not scale well
- Limited automation

Symbolic Model:

- Authenticated encryption modelled very coarsely
- Traditional representation misses attacks

Attack finding until now was manual effort. Can we do better?

Can we automatically detect the impact of subtle AEAD behavior in security protocols?

Our Approach

Our Approach

1. We collected
   - definitions of AEAD constructions and properties
   - known AEAD attacks on the protocol level

What should we model?

Our Approach

1. We collected
    - definitions of AEAD constructions and properties
    - known AEAD attacks on the protocol level
2. We highlighted the relations of properties and proving the missing ones

What should we model?

Our Approach

1. We collected
    - definitions of AEAD constructions and properties
    - known AEAD attacks on the protocol level
2. We highlighted the relations of properties and proving the missing ones
3. We classified of the known attack vectors

What should we model?

Our Approach

1. We collected
   - definitions of AEAD constructions and properties
   - known AEAD attacks on the protocol level
2. We highlighted the relations of properties and proving the missing ones
3. We classified of the known attack vectors

What should we model?

4. We developed of multiple (symbolic) models to address the attack vector classes

Model it

Our Approach

1. We collected
   - definitions of AEAD constructions and properties
   - known AEAD attacks on the protocol level
2. We highlighted the relations of properties and proving the missing ones
3. We classified of the known attack vectors

What should we model?

4. We developed of multiple (symbolic) models to address the attack vector classes

Model it

5. We conducted case studies to show usefulness and feasibility of the new models
   - All case studies were analyzed completely automatic under all models

Test the model

Classify AEAD notions and attacks

Classify AEAD notions and attacks

Gather relations between the existing AEAD notions and properties …

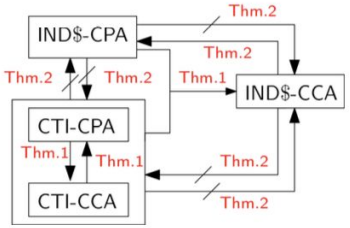
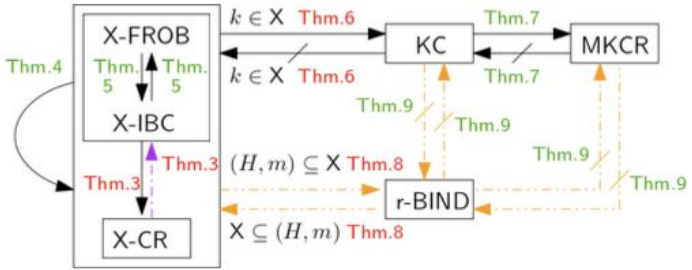


Figure 3: The relation between integrity and privacy for AEAD.



Classify AEAD notions and attacks

Gather relations between the existing AEAD notions and properties …

…and prove the missing ones

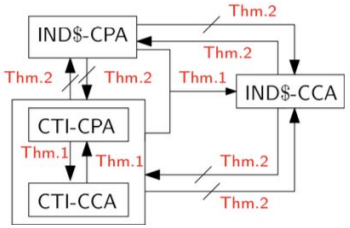
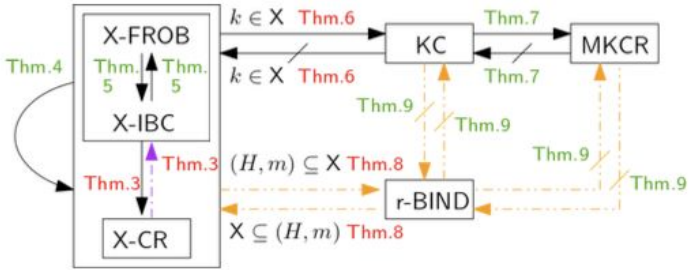


Figure 3: The relation between integrity and privacy for AEAD.



Classify AEAD notions and attacks

Gather relations between the existing AEAD notions and properties …

…and prove the missing ones

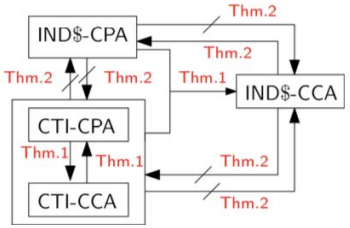
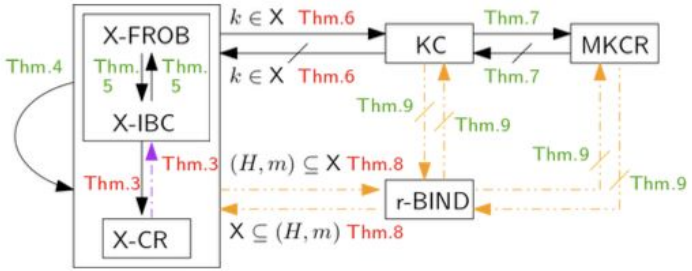


Figure 3: The relation between integrity and privacy for AEAD.

We identify three big theoretical classes, that also allow to capture most practical attacks:

- Integrity & Privacy
- Collision Resistance
- Nonce Reuse



AEAD Security in practice

Concrete AEAD	Integrity and Privacy	Full Collision Resistance	Nonce Misuse Resistance
XSalsa20-Poly1305	●	✗	✗ Xor of plaintexts
AES-GCM	✓	✗	✗ Forgeability + xor of plaintexts
ChaCha20-Poly1305	✓	✗	✗ Xor of plaintexts
OCB3	✓	✗	✗ Forgeability + equality of blocks
EtM (unrelated keys)	✓	✗	✗ Encryption dependent
AES-CCM	✓	●	✗ Xor of plaintexts
AES-EAX	✓	●	✗ Xor of plaintexts
EtM (related keys)	✓	✓	✗ Encryption dependent
CAU-C4	✓	✓	✗ Forgeability + Xor of plaintexts
AES-GCM-SIV	✓	✗	✓
CAU-SIV-C4	✓	✓	✓

✓ : proven in the cited work(s). ● : we conjecture that this holds, but do not know of a proof.
✗ : does not hold, with reference or explanation of counterexample.

Our theoretical models of AEAD weaknesses

Weaknesses in the main classes:

- Integrity & Privacy weakness
- Collisions
- Nonce Reuse

Our theoretical models of AEAD weaknesses

Weaknesses in the main classes:

- Integrity & Privacy weakness
- Collisions
- Nonce Reuse

For completion

Additional AEAD misuses:

- Decryption Misuse
- Tag Misuse
- Commit

Our theoretical models of AEAD weaknesses

Weaknesses in the main classes:

- Integrity & Privacy weakness
- Collisions
- Nonce Reuse

For completion

Additional AEAD misuses:

- Decryption Misuse
- Tag Misuse
- Commit

Each weakness (class)

- has potentially multiple variants
- is modelled as an attacker capability
- can be combined in arbitrary fashion with the other classes

Our theoretical models of AEAD weaknesses

Weaknesses in the main classes:

- Integrity & Privacy weakness
- Collisions
- Nonce Reuse

For completion

Additional AEAD misuses:

- Decryption Misuse
- Tag Misuse
- Commit

Each weakness (class)

- has potentially multiple variants
- is modelled as an attacker capability
- can be combined in arbitrary fashion with the other classes

Let's put the models in practice: the Tamarin Prover 

Case Study Methodology: Two Approaches

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Case Study Methodology: Two Approaches

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Targeted Approach:

Check the protocol in the closest scenario from the real world

Preemptive Approach:

Check the protocol in all possible AEAD threat models

Case Study Methodology: Two Approaches

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Targeted Approach:

Check the protocol in the closest scenario from the real world

Suitable for protocol analysis if:

- the concrete AEAD construction is known

Case Study Methodology: Two Approaches

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Targeted Approach:

Check the protocol in the closest scenario from the real world, by extracting the info from the real world (in)-security of the concrete AEAD scheme used (see table)

Suitable for protocol analysis if:

- the concrete AEAD construction is known

Concrete AEAD	Integrity and Privacy	Full Collision Resistance	Nonce Misuse Resistance
XSalsa20-Poly1305	•	✗	✗ Xor of plaintexts
AES-GCM	✓	✗	✗ Forgeability + xor of plaintexts
ChaCha20-Poly1305	✓	✗	✗ Xor of plaintexts
OCB3	✓	✗	✗ Forgeability + equality of blocks
EtM (unrelated keys)	✓	✗	✗ Encryption dependent
AES-CCM	✓	•	✗ Xor of plaintexts
AES-EAX	✓	•	✗ Xor of plaintexts
EtM (related keys)	✓	✓	✗ Encryption dependent
CAU-C4	✓	✓	✗ Forgeability + Xor of plaintexts
AES-GCM-SIV	✓	✗	✓
CAU-SIV-C4	✓	✓	✓

✓ : proven in the cited work(s). • : we conjecture that this holds, but do not know of a proof.
✗ : does not hold, with reference or explanation of counterexample.

Case Study Methodology: Two Approaches

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Targeted Approach:

Check the protocol in the closest scenario from the real world, by extracting the info from the real world (in)-security of the concrete AEAD scheme used (see table)

Suitable for protocol analysis if:

- the concrete AEAD construction is known

Results:

- Is there currently an attack on the protocol?

Concrete AEAD	Integrity and Privacy	Full Collision Resistance	Nonce Misuse Resistance
XSalsa20-Poly1305	•	✗	✗ Xor of plaintexts
AES-GCM	✓	✗	✗ Forgeability + xor of plaintexts
ChaCha20-Poly1305	✓	✗	✗ Xor of plaintexts
OCB3	✓	✗	✗ Forgeability + equality of blocks
EtM (unrelated keys)	✓	✗	✗ Encryption dependent
AES-CCM	✓	•	✗ Xor of plaintexts
AES-EAX	✓	•	✗ Xor of plaintexts
EtM (related keys)	✓	✓	✗ Encryption dependent
CAU-C4	✓	✓	✗ Forgeability + Xor of plaintexts
AES-GCM-SIV	✓	✗	✓
CAU-SIV-C4	✓	✓	✓

✓ : proven in the cited work(s). • : we conjecture that this holds, but do not know of a proof.
✗ : does not hold, with reference or explanation of counterexample.

Case Study Methodology: Two Approaches

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Preemptive Approach:

Check the protocol in all possible AEADs threat models

Suitable for protocol analysis if:

- one wants to find the requirements of the AEAD for a given protocol

Case Study Methodology: Two Approaches

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

AEAD_Wrapper(Model):

Run all combinations automatically and report the results

Preemptive Approach:

Check the protocol in all possible AEADs threat models

Suitable for protocol analysis if:

- one wants to find the requirements of the AEAD for a given protocol

Case Study Methodology: Two Approaches

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

AEAD_Wrapper(Model):

Run all combinations automatically and report the results

Preemptive Approach:

Check the protocol in all possible AEADs threat models

Suitable for protocol analysis if:

- one wants to find the requirements of the AEAD for a given protocol

Results:

- Minimal threat models that lead to potential attack
- Strongest threat models under which the protocol remains secure

Case Studies: Targeted Approach

Protocol	YubiHSM	SFrame	FB Message Franking
Attacked property	Key Secrecy	Authentication	Reporting
AEAD instance	AES-CCM	AES-GCM, EtM CTR	AES-GCM
Attack Model	Nonce Misuse	Tag	Collision
Time	2s	<1s	1s

Case Studies: Preemptive Approach

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Content agreement: Do all people within a group see the same set of messages?

Protocol	GPG SED	GPG SEIPDv2	Saltpack	Web Push API	WhatsApp	Scuttlebutt
Property	Content Agreement	Content Agreement	Content Agreement	Server Accountability	Content Agreement	Content Agreement
AEAD instance	PGP-CFB	AES-OCB	XSalsa20-Poly1305	AES-GCM	EtM CBC	XSalsa20-Poly1305
Assigned Class	Collision	Collision	Collision	Collision	Collision	Collision
Status	Yes, but deprecated	Infeasible	Infeasible	Reported	Reported	Reported

Case Studies: Preemptive Approach

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Content agreement: Do all people within a group see the same set of messages?

Protocol	GPG SED	GPG SEIPDv2	Saltpack	Web Push API	WhatsApp	Scuttlebutt
Property	Content Agreement	Content Agreement	Content Agreement	Server Accountability	Content Agreement	Content Agreement
AEAD instance	PGP-CFB	AES-OCB	XSalsa20-Poly1305	AES-GCM	EtM CBC	XSalsa20-Poly1305
Assigned Class	Collision	Collision	Collision	Collision	Collision	Collision
Status	Yes, but deprecated	Infeasible	Infeasible	Reported	Reported	Reported

The full automated Tamarin analysis took less than 2 hours!

Automated Analysis of Protocols that use Authenticated Encryption:

Analysing the Impact of the Subtle Differences between AEADs on Protocol Security

Thanks for *another* Distinguished Paper Award!

Alexander Dax: alexander.dax@cispa.de

Artifact: https://github.com/AutomatedAnalysisOf/AEADProtocols

Paper: https://inria.hal.science/hal-04126116v2

References

[KS12] Künnemann, R., & Steel, G. (2013). YubiSecure? Formal security analysis results for the Yubikey and YubiHSM. In *Security and Trust Management: 8th International Workshop, STM 2012, Pisa, Italy, September 13-14, 2012, Revised Selected Papers 8* (pp. 257-272). Springer Berlin Heidelberg.

[IIM21] Isobe, T., Ito, R., & Minematsu, K. (2021). Security Analysis of SFrame. In *Computer Security–ESORICS 2021: 26th European Symposium on Research in Computer Security, Darmstadt, Germany, October 4–8, 2021, Proceedings, Part II 26* (pp. 127-146). Springer International Publishing.

[DGRW18] Dodis, Y., Grubbs, P., Ristenpart, T., & Woodage, J. (2018). Fast message franking: From invisible salamanders to encryptment. In *Advances in Cryptology–CRYPTO 2018: 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2018, Proceedings, Part I 38* (pp. 155-186). Springer International Publishing.