obtaining online sprp security through an optimal inverse-free construction

DIAC 2015, singapore

ritam bhaumik and mridul nandi

indian statistical institute, kolkata

29 september 2015

pseudorandomness

pseudorandomness

We want to **sample uniformly** from a family of functions, but it is **prohibitively large**.

pseudorandomness

We want to **sample uniformly** from a family of functions, but it is **prohibitively large**.

Suppose we find a **very small subfamily** such that it is very difficult to distinguish between **sampling uniformly from this subfamily** and **sampling uniformly from the larger family**. Then we can sample a uniform member of this subfamily and use it as a **representative of the larger family**.

pseudorandomness

We want to **sample uniformly** from a family of functions, but it is **prohibitively large**.

Suppose we find a **very small subfamily** such that it is very difficult to distinguish between **sampling uniformly from this subfamily** and **sampling uniformly from the larger family**. Then we can sample a uniform member of this subfamily and use it as a **representative of the larger family**.

Such a small subfamily of functions (usually indexed by a key, to make sampling convenient) is said to be **pseudorandom** in the larger family.

pseudorandomness

We want to **sample uniformly** from a family of functions, but it is **prohibitively large**.

Suppose we find a **very small subfamily** such that it is very difficult to distinguish between **sampling uniformly from this subfamily** and **sampling uniformly from the larger family**. Then we can sample a uniform member of this subfamily and use it as a **representative of the larger family**.

Such a small subfamily of functions (usually indexed by a key, to make sampling convenient) is said to be **pseudorandom** in the larger family.

distinguishing games

distinguishing games

pseudorandomness distinguishing game: A **real oracle** mimics a **random member from the subfamily**; an **ideal oracle** mimics a **random member from the bigger family**. An adversary makes a **limited number of queries** to try and distinguish between the two.

distinguishing games

pseudorandomness distinguishing game: A **real oracle** mimics a **random member from the subfamily**; an **ideal oracle** mimics a **random member from the bigger family**. An adversary makes a **limited number of queries** to try and distinguish between the two.

strong pseudorandomness distinguishing game (only for a family of invertible functions): A **pair of real oracles** mimics a random member from the subfamily **and its inverse**; a **pair of ideal oracles** mimics a random member from the bigger family **and its inverse**. An adversary makes a **limited number of queries** to try and distinguish between the two pairs.

distinguishing games

pseudorandomness distinguishing game: A **real oracle** mimics a **random member from the subfamily**; an **ideal oracle** mimics a **random member from the bigger family**. An adversary makes a **limited number of queries** to try and distinguish between the two.

strong pseudorandomness distinguishing game (only for a family of invertible functions): A **pair of real oracles** mimics a random member from the subfamily **and its inverse**; a **pair of ideal oracles** mimics a random member from the bigger family **and its inverse**. An adversary makes a **limited number of queries** to try and distinguish between the two pairs.

two basic security notions

two basic security notions

sprp: a family of **permutations** indistinguishable from a random permutation in the strong pseudorandomness game.

two basic security notions

sprp: a family of **permutations** indistinguishable from a random permutation in the strong pseudorandomness game.

A permutation is called **online** if it is **length-preserving**, and an **output-prefix of a particular length** depends only on the **input-prefix of the same length**. (When we talk of length in this context, we usually mean number of blocks.)

two basic security notions

sprp: a family of **permutations** indistinguishable from a random permutation in the strong pseudorandomness game.

A permutation is called **online** if it is **length-preserving**, and an **output-prefix of a particular length** depends only on the **input-prefix of the same length**. (When we talk of length in this context, we usually mean number of blocks.)

online ciphers: a family of **online permutations** which does not leak any information about the input beyond **information on common prefixes**.

two basic security notions

sprp: a family of **permutations** indistinguishable from a random permutation in the strong pseudorandomness game.

A permutation is called **online** if it is **length-preserving**, and an **output-prefix of a particular length** depends only on the **input-prefix of the same length**. (When we talk of length in this context, we usually mean number of blocks.)

online ciphers: a family of **online permutations** which does not leak any information about the input beyond **information on common prefixes**.

sprp security and online ciphers

sprp security and online ciphers

Let us quickly review some basic feautres of two common symmetric-key primitives: **strong pseudorandom permutations** and **online ciphers**:

sprp security and online ciphers

Let us quickly review some basic feautres of two common symmetric-key primitives: **strong pseudorandom permutations** and **online ciphers**:

strong pseudorandom permutations

online ciphers

sprp security and online ciphers

Let us quickly review some basic feautres of two common symmetric-key primitives: **strong pseudorandom permutations** and **online ciphers**:

strong pseudorandom permutations

1. **a very strong notion of security**

online ciphers

sprp security and online ciphers

Let us quickly review some basic feautres of two common symmetric-key primitives: **strong pseudorandom permutations** and **online ciphers**:

strong pseudorandom permutations

1. a very strong notion of security
2. **costly to implement**

online ciphers

sprp security and online ciphers

Let us quickly review some basic feautres of two common symmetric-key primitives: **strong pseudorandom permutations** and **online ciphers**:

strong pseudorandom permutations

1. a very strong notion of security
2. costly to implement
3. **requires multiple passes**

online ciphers

sprp security and online ciphers

Let us quickly review some basic feautres of two common symmetric-key primitives: **strong pseudorandom permutations** and **online ciphers**:

strong pseudorandom permutations

1. a very strong notion of security
2. costly to implement
3. requires multiple passes

online ciphers

1. **highly efficient single-pass execution**

sprp security and online ciphers

Let us quickly review some basic feautres of two common symmetric-key primitives: **strong pseudorandom permutations** and **online ciphers**:

strong pseudorandom permutations

1. a very strong notion of security
2. costly to implement
3. requires multiple passes

online ciphers

1. highly efficient single-pass execution
2. **cheap implementation with low buffer size**

sprp security and online ciphers

Let us quickly review some basic feautres of two common symmetric-key primitives: **strong pseudorandom permutations** and **online ciphers**:

strong pseudorandom permutations

1. a very strong notion of security
2. costly to implement
3. requires multiple passes

online ciphers

1. highly efficient single-pass execution
2. cheap implementation with low buffer size
3. **can never achieve sprp security**

sprp security and online ciphers

Let us quickly review some basic feautres of two common symmetric-key primitives: **strong pseudorandom permutations** and **online ciphers**:

strong pseudorandom permutations

1. a very strong notion of security
2. costly to implement
3. requires multiple passes

online ciphers

1. highly efficient single-pass execution
2. cheap implementation with low buffer size
3. can never achieve sprp security

the notion of online sprp security

the notion of online sprp security

It is well-established that if we are to achieve the efficiency of **online ciphers**, we cannot hope to maintain **sprp security**. Hence, we shall now minimally dilute the notion of sprp security to obtain what we call **online sprp security:**

the notion of online sprp security

It is well-established that if we are to achieve the efficiency of **online ciphers**, we cannot hope to maintain **sprp security**. Hence, we shall now minimally dilute the notion of sprp security to obtain what we call **online sprp security:**

definition (online sprp security)

the notion of online sprp security

It is well-established that if we are to achieve the efficiency of **online ciphers**, we cannot hope to maintain **sprp security**. Hence, we shall now minimally dilute the notion of sprp security to obtain what we call **online sprp security:**

definition (online sprp security)

A family of online permutations is said to have **online sprp security** if an adversary with access to both encryption and decryption oracles cannot distinguish a uniformly chosen member of the family from a uniformly chosen online permutation. In other words it is **strong pseudorandom** in the family of all online permutations.

the notion of online sprp security

It is well-established that if we are to achieve the efficiency of **online ciphers**, we cannot hope to maintain **sprp security**. Hence, we shall now minimally dilute the notion of sprp security to obtain what we call **online sprp security:**

definition (online sprp security)

A family of online permutations is said to have **online sprp security** if an adversary with access to both encryption and decryption oracles cannot distinguish a uniformly chosen member of the family from a uniformly chosen online permutation. In other words it is **strong pseudorandom** in the family of all online permutations.

inverse-free constructions

inverse-free constructions

Blockcipher-based encryption schemes often use the **inverse of the underlying blockcipher** for decryption. This has certain drawbacks:

inverse-free constructions

Blockcipher-based encryption schemes often use the **inverse of the underlying blockcipher** for decryption. This has certain drawbacks:

- in a **combined implementation**, the footprint size goes up;

inverse-free constructions

Blockcipher-based encryption schemes often use the **inverse of the underlying blockcipher** for decryption. This has certain drawbacks:

- in a **combined implementation**, the footprint size goes up;
- the underlying blockcipher is required to be **sprp secure**;

inverse-free constructions

Blockcipher-based encryption schemes often use the **inverse of the underlying blockcipher** for decryption. This has certain drawbacks:

- in a **combined implementation**, the footprint size goes up;
- the underlying blockcipher is required to be **sprp secure**;
- decryption of underlying blockcipher is often **costlier**.

inverse-free constructions

Blockcipher-based encryption schemes often use the **inverse of the underlying blockcipher** for decryption. This has certain drawbacks:

- in a **combined implementation**, the footprint size goes up;
- the underlying blockcipher is required to be **sprp secure**;
- decryption of underlying blockcipher is often **costlier**.

definition (inverse-free encryption schemes)

inverse-free constructions

Blockcipher-based encryption schemes often use the **inverse of the underlying blockcipher** for decryption. This has certain drawbacks:

- in a **combined implementation**, the footprint size goes up;
- the underlying blockcipher is required to be **sprp secure**;
- decryption of underlying blockcipher is often **costlier**.

---

definition (inverse-free encryption schemes)

An **inverse-free encryption scheme** is one that does not call the inverse of the underlying blockcipher for either encryption or decryption.

inverse-free constructions

Blockcipher-based encryption schemes often use the **inverse of the underlying blockcipher** for decryption. This has certain drawbacks:

- in a **combined implementation**, the footprint size goes up;
- the underlying blockcipher is required to be **sprp secure**;
- decryption of underlying blockcipher is often **costlier**.

definition (inverse-free encryption schemes)

An **inverse-free encryption scheme** is one that does not call the inverse of the underlying blockcipher for either encryption or decryption.

feistel networks

feistel networks

Inverse-free constructions originate in the very elegant networks first devised by **Horst Feistel** and famously analysed for security by **Luby and Rackoff**. Almost all inverse-free constructions so far have built on the basic idea of feistel networks.

feistel networks

Inverse-free constructions originate in the very elegant networks first devised by **Horst Feistel** and famously analysed for security by **Luby and Rackoff**. Almost all inverse-free constructions so far have built on the basic idea of feistel networks.

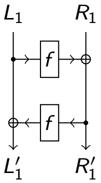


Figure : 2-round feistel encryption

Illustrated are encryption and decryption diagrams for two rounds of feistel. Here, f is an **ideal random function** which is generally implemented by a **blockcipher**.

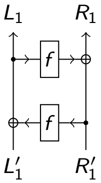


Figure : 2-round feistel decryption

feistel networks

Inverse-free constructions originate in the very elegant networks first devised by **Horst Feistel** and famously analysed for security by **Luby and Rackoff**. Almost all inverse-free constructions so far have built on the basic idea of feistel networks.

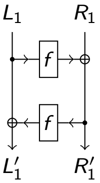


Figure : 2-round feistel encryption

Illustrated are encryption and decryption diagrams for two rounds of feistel. Here, f is an **ideal random function** which is generally implemented by a **blockcipher**.

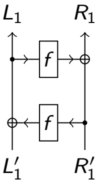


Figure : 2-round feistel decryption

online AND inverse-free?

online AND inverse-free?

It is tempting to envisage an **online cipher** that is also **inverse-free**. However, here we run into a problem.

online AND inverse-free?

It is tempting to envisage an **online cipher** that is also **inverse-free**. However, here we run into a problem.

problem

online AND inverse-free?

It is tempting to envisage an **online cipher** that is also **inverse-free**. However, here we run into a problem.

problem

No known inverse-free construction can incorporate **single-block inputs**.

online AND inverse-free?

It is tempting to envisage an **online cipher** that is also **inverse-free**. However, here we run into a problem.

problem

No known inverse-free construction can incorporate **single-block inputs**.

In other words, an online cipher **in the conventional sense** cannot use one of the **known inverse-free designs**. Indeed, we believe no such inverse-free construction exists.

online AND inverse-free?

It is tempting to envisage an **online cipher** that is also **inverse-free**. However, here we run into a problem.

problem

No known inverse-free construction can incorporate **single-block inputs**.

In other words, an online cipher **in the conventional sense** cannot use one of the **known inverse-free designs**. Indeed, we believe no such inverse-free construction exists.

diblocks

diblocks

Fortunately, we can cheat our way around this obstacle, using **diblocks.**

diblocks

Fortunately, we can cheat our way around this obstacle, using **diblocks.**

definition (diblock)

diblocks

Fortunately, we can cheat our way around this obstacle, using **diblocks.**

---

definition (diblock)

An odd block and the even block immediately following it together constitute what we call a **diblock**.

diblocks

Fortunately, we can cheat our way around this obstacle, using **diblocks.**

definition (diblock)

An odd block and the even block immediately following it together constitute what we call a **diblock**.

definition (diblock-online)

diblocks

Fortunately, we can cheat our way around this obstacle, using **diblocks.**

definition (diblock)

An odd block and the even block immediately following it together constitute what we call a **diblock**.

definition (diblock-online)

A permutation is said to be **diblock-online** if for $i = 1, 2, \ldots,$ the first $2i$ output blocks depend only on the first $2i$ input blocks.

diblocks

Fortunately, we can cheat our way around this obstacle, using **diblocks.**

definition (diblock)

An odd block and the even block immediately following it together constitute what we call a **diblock**.

definition (diblock-online)

A permutation is said to be **diblock-online** if for $i = 1, 2, \ldots,$ the first $2i$ output blocks depend only on the first $2i$ input blocks.

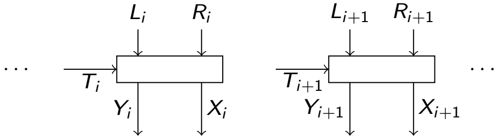
a schematic view of OleF

a schematic view of OIeF

We now present a schematic view of OIeF.

a schematic view of OleF

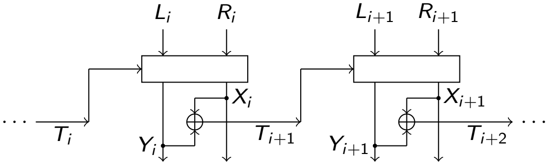
We now present a schematic view of OleF.



layer 1: sequential encryption, based on 2-round feistel

a schematic view of OleF

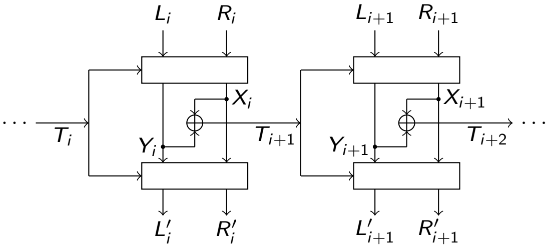
We now present a schematic view of OleF.



layer 2: mixing and generating tweak for next diblock

a schematic view of OleF

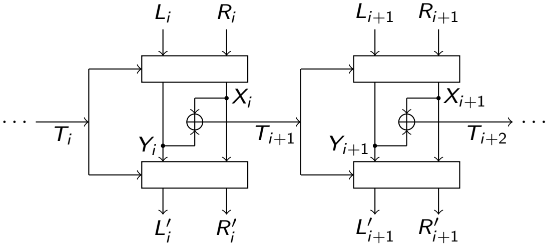
We now present a schematic view of OleF.



layer 3: parallel encryption, based on 2-round feistel

a schematic view of OleF

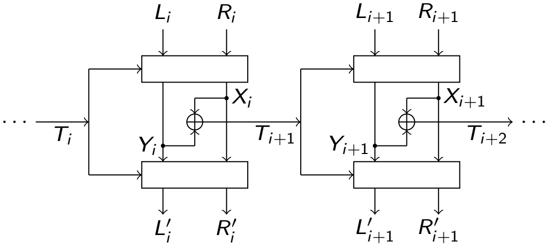
We now present a schematic view of OleF.



overall paradigm: **encrypt-mix-encrypt**

a schematic view of OleF

We now present a schematic view of OleF.



overall paradigm: **encrypt-mix-encrypt**

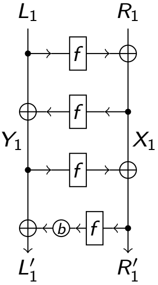
The complete OleF construction

The complete OleF construction

We're now ready to present the complete construction of OleF, for a 2ℓ-block input.

The complete OleF construction

We're now ready to present the complete construction of OleF, for a 2ℓ-block input.

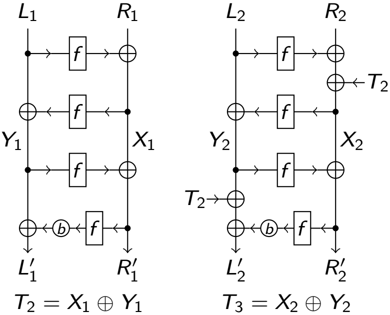


$T_2 = X_1 \oplus Y_1$

input-diblock 1 is processed to obtain **output-diblock** 1 and **tweak** T_2

The complete OleF construction

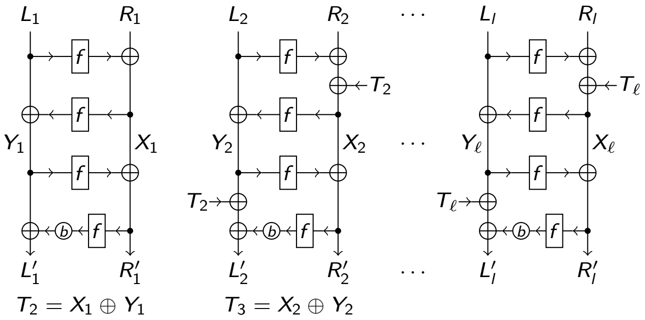
We're now ready to present the complete construction of OleF, for a 2ℓ-block input.



input-diblock 2 is processed using T_2 to obtain **output-diblock** 2 and **tweak** T_3

The complete OleF construction

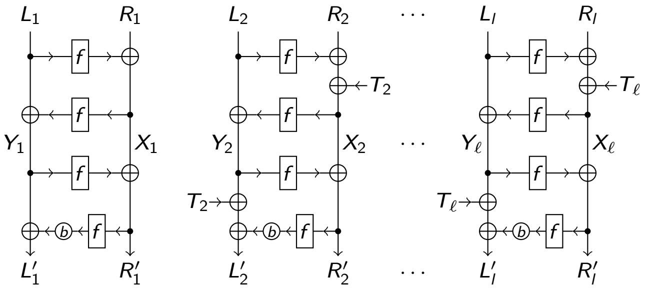
We're now ready to present the complete construction of OleF, for a 2ℓ-block input.



input-diblock ℓ is processed last using T_ℓ to obtain **output-diblock** ℓ

The complete OleF construction

We're now ready to present the complete construction of OleF, for a 2ℓ-block input.

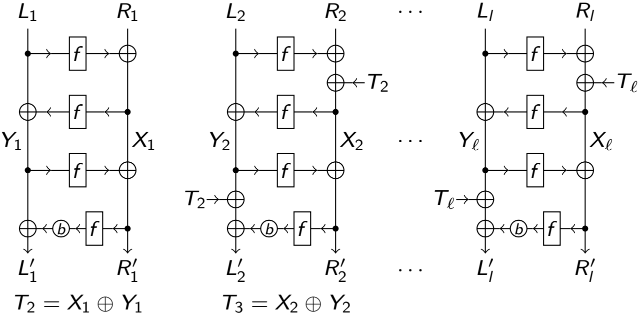


$T_2 = X_1 \oplus Y_1$ $\quad$ $T_3 = X_2 \oplus Y_2$

4ℓ calls in all to the underlying blockcipher

The complete OleF construction

We're now ready to present the complete construction of OleF, for a 2ℓ-block input.



4ℓ calls in all to the underlying blockcipher

handling partial diblocks

handling partial diblocks

Next we take a look at how incomplete diblocks are processed.

handling partial diblocks

Next we take a look at how incomplete diblocks are processed.

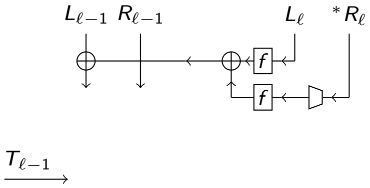
$$\begin{array}{cccc} L_{\ell-1} & R_{\ell-1} & L_\ell & {}^*R_\ell \\ \downarrow & \downarrow & \downarrow & \downarrow \end{array}$$

1. suppose ${}^*R_\ell$ is an incomplete block

$$\xrightarrow{T_{\ell-1}}$$

handling partial diblocks

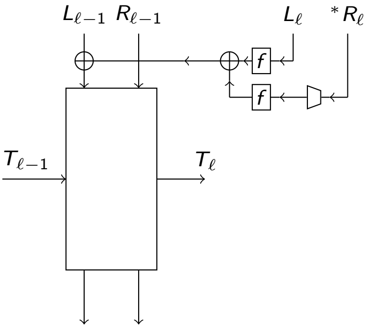
Next we take a look at how incomplete diblocks are processed.



1. suppose $*R_\ell$ is an incomplete block
2. we add to $L_{\ell-1}$ information on L_ℓ and $*R_\ell$

handling partial diblocks

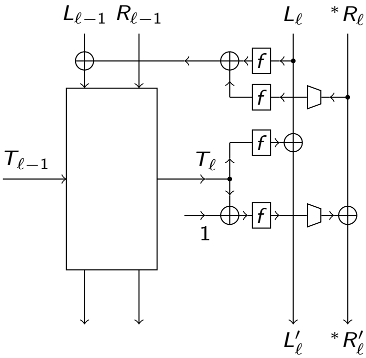
Next we take a look at how incomplete diblocks are processed.



1. suppose $^*R_\ell$ is an incomplete block
2. we add to $L_{\ell-1}$ information on L_ℓ and $^*R_\ell$
3. we process the modified $L_{\ell-1}$ and $R_{\ell-1}$ normally using $T_{\ell-1}$, to obtain T_ℓ

handling partial diblocks

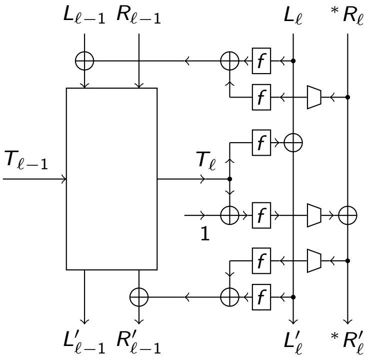
Next we take a look at how incomplete diblocks are processed.



1. suppose $*R_\ell$ is an incomplete block
2. we add to $L_{\ell-1}$ information on L_ℓ and $*R_\ell$
3. we process the modified $L_{\ell-1}$ and $R_{\ell-1}$ normally using $T_{\ell-1}$, to obtain T_ℓ
4. we use T_ℓ in counter mode to obtain L'_ℓ and $*R'_\ell$

handling partial diblocks

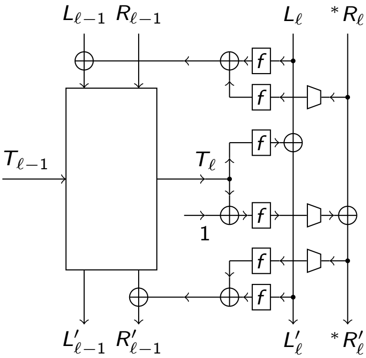
Next we take a look at how incomplete diblocks are processed.



1. suppose $^*R_\ell$ is an incomplete block
2. we add to $L_{\ell-1}$ information on L_ℓ and $^*R_\ell$
3. we process the modified $L_{\ell-1}$ and $R_{\ell-1}$ normally using $T_{\ell-1}$, to obtain T_ℓ
4. we use T_ℓ in counter mode to obtain L'_ℓ and $^*R'_\ell$
5. we use L'_ℓ and $^*R'_\ell$ to obtain $L'_{\ell-1}$ and $R'_{\ell-1}$

handling partial diblocks

Next we take a look at how incomplete diblocks are processed.



1. suppose $^*R_\ell$ is an incomplete block
2. we add to $L_{\ell-1}$ information on L_ℓ and $^*R_\ell$
3. we process the modified $L_{\ell-1}$ and $R_{\ell-1}$ normally using $T_{\ell-1}$, to obtain T_ℓ
4. we use T_ℓ in counter mode to obtain L'_ℓ and $^*R'_\ell$
5. we use L'_ℓ and $^*R'_\ell$ to obtain $L'_{\ell-1}$ and $R'_{\ell-1}$

comparison with similar constructions

comparison with similar constructions

Let's see how OIeF fares against certain similar constructions in inverse-free implementations.

comparison with similar constructions

Let's see how OleF fares against certain similar constructions in inverse-free implementations.

construction	f-calls per diblock	online?
MCBC	7	yes
TC3	6	yes
AEZ	5	no
FMix	4	no
OleF	4	yes

Table : comparing calls to f per diblock for various constructions in inverse-free implementations

comparison with similar constructions

Let's see how OleF fares against certain similar constructions in inverse-free implementations.

construction	f-calls per diblock	online?
MCBC	7	yes
TC3	6	yes
AEZ	5	no
FMix	4	no
OleF	4	yes

Table : comparing calls to f per diblock for various constructions in inverse-free implementations

robust authenticated encryption

robust authenticated encryption

For **authenticated encryption schemes** that **release unverified plaintext** even for invalid ciphertexts, one can desire that this released plaintext reveals nothing about the encryption function that can damage the privacy or authenticity of other encryptions. In a recent work, **Huang, Krovatz and Rogaway** have come up with a notion of **robust authenticated encryption**, which can be described by the following distinguishing game:

robust authenticated encryption

For **authenticated encryption schemes** that **release unverified plaintext** even for invalid ciphertexts, one can desire that this released plaintext reveals nothing about the encryption function that can damage the privacy or authenticity of other encryptions. In a recent work, **Huang, Krovatz and Rogaway** have come up with a notion of **robust authenticated encryption**, which can be described by the following distinguishing game:

- the ideal random oracles use a **pseudorandom injective function** for handling encryption and valid decryption queries;

robust authenticated encryption

For **authenticated encryption schemes** that **release unverified plaintext** even for invalid ciphertexts, one can desire that this released plaintext reveals nothing about the encryption function that can damage the privacy or authenticity of other encryptions. In a recent work, **Huang, Krovatz and Rogaway** have come up with a notion of **robust authenticated encryption**, which can be described by the following distinguishing game:

- the ideal random oracles use a **pseudorandom injective function** for handling encryption and valid decryption queries;
- the ideal decryption oracle uses a **simulator** for handling invalid decryption queries;

robust authenticated encryption

For **authenticated encryption schemes** that **release unverified plaintext** even for invalid ciphertexts, one can desire that this released plaintext reveals nothing about the encryption function that can damage the privacy or authenticity of other encryptions. In a recent work, **Huang, Krovatz and Rogaway** have come up with a notion of **robust authenticated encryption**, which can be described by the following distinguishing game:

- the ideal random oracles use a **pseudorandom injective function** for handling encryption and valid decryption queries;
- the ideal decryption oracle uses a **simulator** for handling invalid decryption queries;
- the simulator **mimics the distribution of unverified plaintext** as would be released by the real oracle.

robust authenticated encryption

For **authenticated encryption schemes** that **release unverified plaintext** even for invalid ciphertexts, one can desire that this released plaintext reveals nothing about the encryption function that can damage the privacy or authenticity of other encryptions. In a recent work, **Huang, Krovatz and Rogaway** have come up with a notion of **robust authenticated encryption**, which can be described by the following distinguishing game:

- the ideal random oracles use a **pseudorandom injective function** for handling encryption and valid decryption queries;
- the ideal decryption oracle uses a **simulator** for handling invalid decryption queries;
- the simulator **mimics the distribution of unverified plaintext** as would be released by the real oracle.

robust authenticated encryption game

robust authenticated encryption game

Let's take a closer look at the robust authenticated encryption game.

robust authenticated encryption game

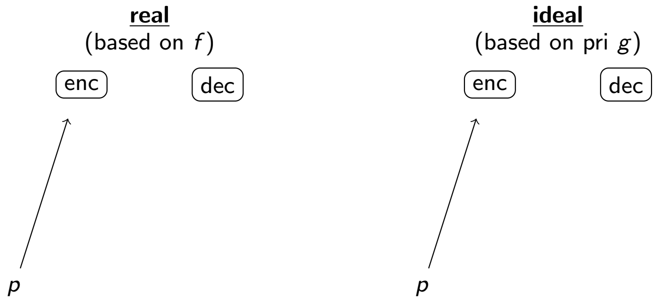
Let's take a closer look at the robust authenticated encryption game.

real (based on f)		**ideal** (based on pri g)	
$\boxed{\text{enc}}$	$\boxed{\text{dec}}$	$\boxed{\text{enc}}$	$\boxed{\text{dec}}$

the ideal oracle first chooses a **pseudorandom injection** g

robust authenticated encryption game

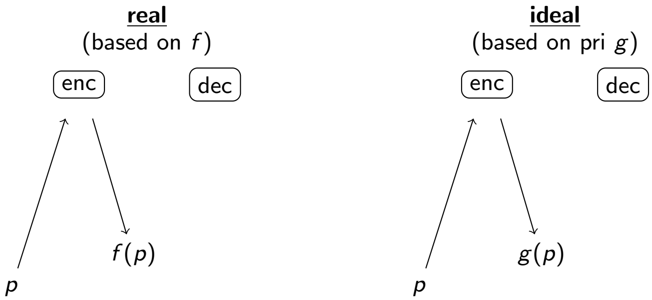
Let's take a closer look at the robust authenticated encryption game.



for a **plaintext** p it just outputs $g(p)$

robust authenticated encryption game

Let's take a closer look at the robust authenticated encryption game.



for a **plaintext** p it just outputs $g(p)$

robust authenticated encryption game

Let's take a closer look at the robust authenticated encryption game.

real
(based on f)

$\boxed{\text{enc}}$ $\boxed{\text{dec}}$

$c \in \text{range}(f)$

ideal
(based on pri g)

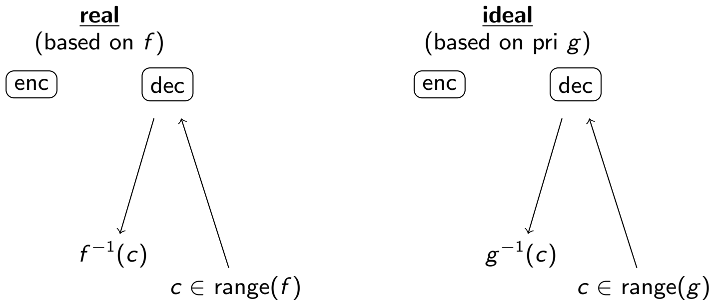
$\boxed{\text{enc}}$ $\boxed{\text{dec}}$

$c \in \text{range}(g)$

for a **valid ciphertext** c it outputs $g^{-1}(p)$

robust authenticated encryption game

Let's take a closer look at the robust authenticated encryption game.



for a **valid ciphertext** c it outputs $g^{-1}(p)$

robust authenticated encryption game

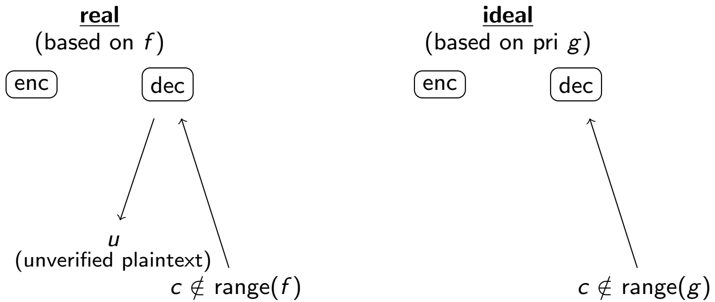
Let's take a closer look at the robust authenticated encryption game.

real (based on f)		**ideal** (based on pri g)	
$\boxed{\text{enc}}$	$\boxed{\text{dec}}$	$\boxed{\text{enc}}$	$\boxed{\text{dec}}$
	$\uparrow$ $c \notin \text{range}(f)$		$\uparrow$ $c \notin \text{range}(g)$

for an **invalid ciphertext** c it **simulates the distribution**
of the unverified plaintext that the real oracle would release

robust authenticated encryption game

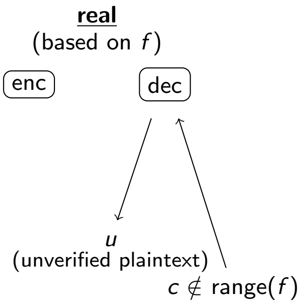
Let's take a closer look at the robust authenticated encryption game.



for an **invalid ciphertext** c it **simulates the distribution** of the unverified plaintext that the real oracle would release

robust authenticated encryption game

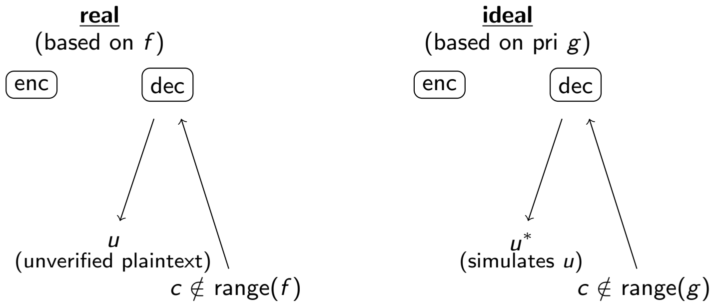
Let's take a closer look at the robust authenticated encryption game.



for an **invalid ciphertext** c it **simulates the distribution** of the unverified plaintext that the real oracle would release

robust authenticated encryption game

Let's take a closer look at the robust authenticated encryption game.



for an **invalid ciphertext** c it **simulates the distribution** of the unverified plaintext that the real oracle would release

online robust authenticated encryption

online robust authenticated encryption

In analogy to robust authenticated encryption schemes, we can define **online robust authenticated encryption schemes**, where the ideal random oracles use a **pseudorandom online injective function**, and the simulator works just as before.

online robust authenticated encryption

In analogy to robust authenticated encryption schemes, we can define **online robust authenticated encryption schemes**, where the ideal random oracles use a **pseudorandom online injective function**, and the simulator works just as before.

Since OleF can handle **variable length inputs**, it can be used in an **encode-then-encipher** framework to obtain a robust online authenticated encryption scheme.

online robust authenticated encryption

In analogy to robust authenticated encryption schemes, we can define **online robust authenticated encryption schemes**, where the ideal random oracles use a **pseudorandom online injective function**, and the simulator works just as before.

Since OleF can handle **variable length inputs**, it can be used in an **encode-then-encipher** framework to obtain a robust online authenticated encryption scheme.

advantages of OleF

advantages of OleF

In summary, OleF has the following advantages:

advantages of OleF

In summary, OleF has the following advantages:

- being **inverse-free**, this construction has three advantages:

advantages of OleF

In summary, OleF has the following advantages:

- being **inverse-free**, this construction has four advantages:
  - a **combined implementation** of encryption and decryption keeps the **footprint low**;

advantages of OleF

In summary, OleF has the following advantages:

- being **inverse-free**, this construction has four advantages:
  - a **combined implementation** of encryption and decryption keeps the **footprint low**;
  - when using certain blockciphers like AES, where **decryption is costlier than encryption**, the **overall cost decreases**;

advantages of OleF

In summary, OleF has the following advantages:

- being **inverse-free**, this construction has four advantages:
  - a **combined implementation** of encryption and decryption keeps the **footprint low**;
  - when using certain blockciphers like AES, where **decryption is costlier than encryption**, the **overall cost decreases**;
  - underlying block function **does not have to be invertible**;

advantages of OleF

In summary, OleF has the following advantages:

- being **inverse-free**, this construction has four advantages:
  - a **combined implementation** of encryption and decryption keeps the **footprint low**;
  - when using certain blockciphers like AES, where **decryption is costlier than encryption**, the **overall cost decreases**;
  - underlying block function **does not have to be invertible**;
  - even if a blockcipher is used, it only needs to be **prp secure**, instead of sprp secure;

advantages of OleF

In summary, OleF has the following advantages:

- being **inverse-free**, this construction has four advantages:
  - a **combined implementation** of encryption and decryption keeps the **footprint low**;
  - when using certain blockciphers like AES, where **decryption is costlier than encryption**, the **overall cost decreases**;
  - underlying block function **does not have to be invertible**;
  - even if a blockcipher is used, it only needs to be **prp secure**, instead of sprp secure;
- being **online**, this is **easier to implement** (due to a **low buffer size**) and also **performs better**;

advantages of OleF

In summary, OleF has the following advantages:

- being **inverse-free**, this construction has four advantages:
  - a **combined implementation** of encryption and decryption keeps the **footprint low**;
  - when using certain blockciphers like AES, where **decryption is costlier than encryption**, the **overall cost decreases**;
  - underlying block function **does not have to be invertible**;
  - even if a blockcipher is used, it only needs to be **prp secure**, instead of sprp secure;
- being **online**, this is **easier to implement** (due to a **low buffer size**) and also **performs better**;
- we believe this is an **optimal inverse-free online sprp** construction, in terms of **the number of calls to the underlying prf**;

advantages of OleF

In summary, OleF has the following advantages:

- being **inverse-free**, this construction has four advantages:
  - a **combined implementation** of encryption and decryption keeps the **footprint low**;
  - when using certain blockciphers like AES, where **decryption is costlier than encryption**, the **overall cost decreases**;
  - underlying block function **does not have to be invertible**;
  - even if a blockcipher is used, it only needs to be **prp secure**, instead of sprp secure;
- being **online**, this is **easier to implement** (due to a **low buffer size**) and also **performs better**;
- we believe this is an **optimal inverse-free online sprp** construction, in terms of **the number of calls to the underlying prf**;
- since this can handle **variable length inputs**, it can be used to obtain a **robust online authenticated encryption** scheme.

advantages of OleF

In summary, OleF has the following advantages:

- being **inverse-free**, this construction has four advantages:
  - a **combined implementation** of encryption and decryption keeps the **footprint low**;
  - when using certain blockciphers like AES, where **decryption is costlier than encryption**, the **overall cost decreases**;
  - underlying block function **does not have to be invertible**;
  - even if a blockcipher is used, it only needs to be **prp secure**, instead of sprp secure;
- being **online**, this is **easier to implement** (due to a **low buffer size**) and also **performs better**;
- we believe this is an **optimal inverse-free online sprp** construction, in terms of **the number of calls to the underlying prf**;
- since this can handle **variable length inputs**, it can be used to obtain a **robust online authenticated encryption** scheme.

that's all folks!