Hensel and Grøstl

A Grim(m) Submission to the Snake Oil Crypto Competition

Roberto Avanzi

Candidate for the post of Minister of Procrastination in the
Selection Committee of the Snake Oil Crypto Competition.
Vote on my membership currently being procrastinated.

Requirementssssss

Since the SOCC is the best crypto competition *ever*, any self-respecting cryptographer is morally **required** to submit a cipher.

SOCC **requirement**: Avoid SERPENT-inspired designs.

Solution: avoid reptiles.

Idea: pick a mammal. A *carnivore*.

Rationale: annoy vegetarian cryptographers (hi, Tanja!).

The animal must have a good (FSE) pedigree.

Biham **E**li and **A**nderson **R**oss's FSE 1996 design.

Requirementssssss

Since the SOCC is the best crypto competition *ever*, any self-respecting cryptographer is morally **required** to submit a cipher.

SOCC **requirement**: Avoid SERPENT-inspired designs.

Solution: avoid reptiles.

Idea: pick a mammal. A *carnivore*.

Rationale: annoy vegetarian cryptographers (hi, Tanja!).

The animal must have a good (FSE) pedigree.

Biham **E**li and **A**nderson **R**oss's FSE 1996 design.

Requirementssssss

Since the SOCC is the best crypto competition *ever*, any self-respecting cryptographer is morally **required** to submit a cipher.

SOCC **requirement**: Avoid SERPENT-inspired designs.

Solution: avoid reptiles.

Idea: pick a mammal. A *carnivore*.

Rationale: annoy vegetarian cryptographers (hi, Tanja!).

The animal must have a good (FSE) pedigree.

Biham **E**li and **A**nderson **R**oss's FSE 1996 design.

Requirementssssss

Since the SOCC is the best crypto competition *ever*, any self-respecting cryptographer is morally **required** to submit a cipher.

SOCC **requirement**: Avoid SERPENT-inspired designs.

Solution: avoid reptiles.

Idea: pick a mammal. A *carnivore*.

Rationale: annoy vegetarian cryptographers (hi, Tanja!).

The animal must have a good (FSE) pedigree.

Biham **E**li and **A**nderson **R**oss's FSE 1996 design.

Requirementssssss

Since the SOCC is the best crypto competition *ever*, any self-respecting cryptographer is morally **required** to submit a cipher.

SOCC **requirement**: Avoid SERPENT-inspired designs.

Solution: avoid reptiles.

Idea: pick a mammal. A *carnivore*.

Rationale: annoy vegetarian cryptographers (hi, Tanja!).

The animal must have a good (FSE) pedigree.

Biham **E**li and **A**nderson **R**oss's FSE 1996 design.

Requirementssssss

Since the SOCC is the best crypto competition *ever*, any self-respecting cryptographer is morally **required** to submit a cipher.

SOCC **requirement**: Avoid SERPENT-inspired designs.

Solution: avoid reptiles.

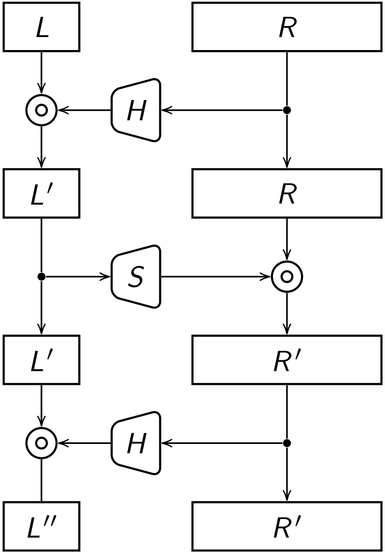
Idea: pick a mammal. A *carnivore*.

Rationale: annoy vegetarian cryptographers (hi, Tanja!).

The animal must have a good (FSE) pedigree.

Biham **E**li and **A**nderson **R**oss's FSE 1996 design.

Desssssign



The hash function H is Grøstl.

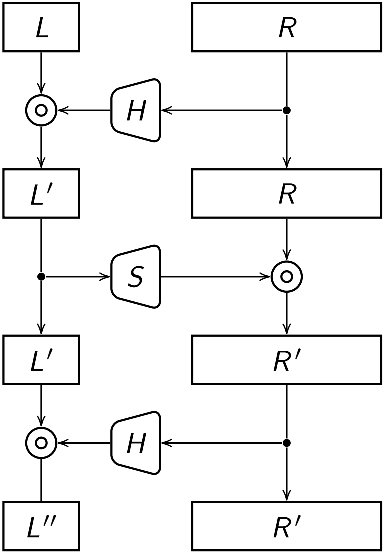
The stream cipher S is as of yet undefined, but we will use Hensel lifting in its design.

Since I will procrastinate its design forever, it will stay secret forever, thus achieving *Perfect Forward Secrecy*.

Hensel and Grøstl!

But: performance problem.
(NO, not Hensel with Grøstl.)

Dessssign



The hash function H is Grøstl.

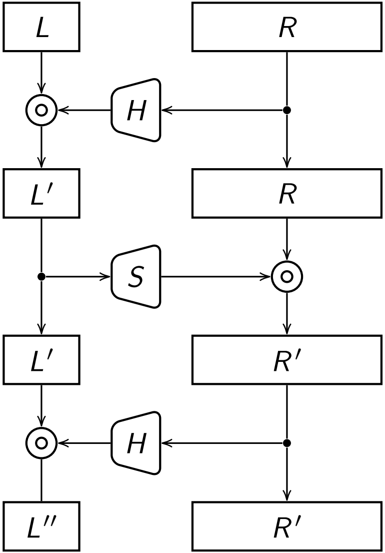
The stream cipher S is as of yet undefined, but we will use Hensel lifting in its design.

Since I will procrastinate its design forever, it will stay secret forever, thus achieving *Perfect Forward Secrecy*.

Hensel and Grøstl!

But: performance problem.
(NO, not Hensel with Grøstl.)

Desssssign



The hash function H is Grøstl.

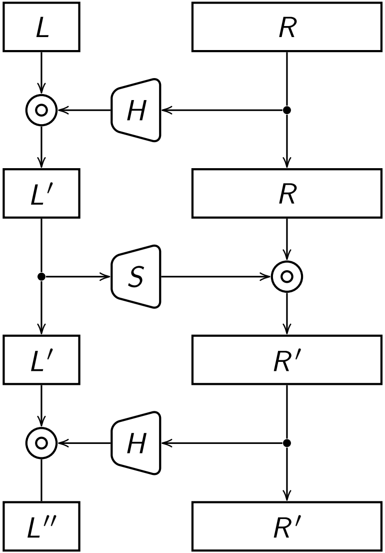
The stream cipher S is as of yet undefined, but we will use Hensel lifting in its design.

Since I will procrastinate its design forever, it will stay secret forever, thus achieving *Perfect Forward Secrecy*.

Hensel and Grøstl!

But: performance problem.
(NO, not Hensel with Grøstl.)

Dessssign



The hash function H is Grøstl.

The stream cipher S is as of yet undefined, but we will use Hensel lifting in its design.

Since I will procrastinate its design forever, it will stay secret forever, thus achieving *Perfect Forward Secrecy*.

Hensel and Grøstl!

But: performance problem.
(NO, not Hensel with Grøstl.)

Optimisssssssationsssssss

How to double the speed of EVERY block cipher!

For each block compute parity.

If 1 encrypt using proposed scheme.

If 0 just XOR the block with a
random number. We use the
same random number as the PS3.
(Widely deployed $\Rightarrow$ well studied.)

It is so secure that we do not know how to consistently decrypt.
(Possible tweak: Consider adding a parity bit to each block?)

Optimisssssssationsssssss

How to double the speed of EVERY block cipher!

For each block compute parity.

If 1 encrypt using proposed scheme.

If 0 just XOR the block with a random number. We use the same random number as the PS3. (Widely deployed ⇒ well studied.)

It is so secure that we do not know how to consistently decrypt. (Possible tweak: Consider adding a parity bit to each block?)

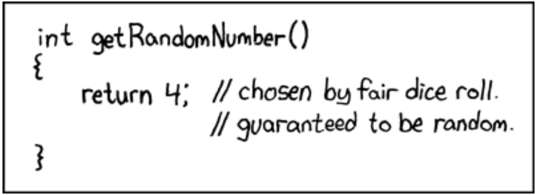
Optimissssssssationsssssss

How to double the speed of EVERY block cipher!

For each block compute parity.

If 1 encrypt using proposed scheme.

If 0 just XOR the block with a random number. We use the same random number as the PS3. (Widely deployed ⇒ well studied.)



```
int getRandomNumber()
{
    return 4;  // chosen by fair dice roll.
               // guaranteed to be random.
}
```

It is so secure that we do not know how to consistently decrypt. (Possible tweak: Consider adding a parity bit to each block?)

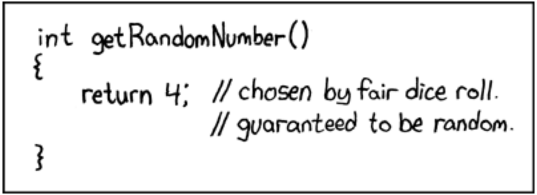
Optimissssssssationssssssss

How to double the speed of EVERY block cipher!

For each block compute parity.

If 1 encrypt using proposed scheme.

If 0 just XOR the block with a random number. We use the same random number as the PS3. (Widely deployed ⇒ well studied.)



```
int getRandomNumber()
{
    return 4;  // chosen by fair dice roll.
               // guaranteed to be random.
}
```

It is so secure that we do not know how to consistently decrypt. (Possible tweak: Consider adding a parity bit to each block?)

Why choossssssse Hensel and Grøstl?

Catchy name; cool logo.

Follows industrial best practices:
security through obscurity.
Component secret even to designer!
Combined with PFS!

Inefficient: Slower attacks!

Reuses RNG by another company.

It is so secure that we do not
know how to decrypt.

Bonus: Jean-Jacques approved design
$\Rightarrow$ **GCHQ analysed it already!**

Why choosssssssse Hensel and Grøstl?

Catchy name; cool logo.

Follows industrial best practices:
security through obscurity.
Component secret even to designer!
Combined with PFS!

Inefficient: Slower attacks!

Reuses RNG by another company.

It is so secure that we do not
know how to decrypt.

Bonus: Jean-Jacques approved design
⇒ GCHQ analysed it already!

Why choosssssssse Hensel and Grøstl?

Catchy name; cool logo.

Follows industrial best practices:
security through obscurity.
Component secret even to designer!
Combined with PFS!

Inefficient: Slower attacks!

Reuses RNG by another company.

It is so secure that we do not
know how to decrypt.

Bonus: Jean-Jacques approved design
$\Rightarrow$ **GCHQ analysed it already!**

Why choossssssse Hensel and Grøstl?

Catchy name; cool logo.

Follows industrial best practices:
security through obscurity.
Component secret even to designer!
Combined with PFS!

Inefficient: Slower attacks!

Reuses RNG by another company.

It is so secure that we do not
know how to decrypt.

Bonus: Jean-Jacques approved design
$\Rightarrow$ **GCHQ analysed it already!**