Dial C for Cipher

Le chiffrement était presque parfait

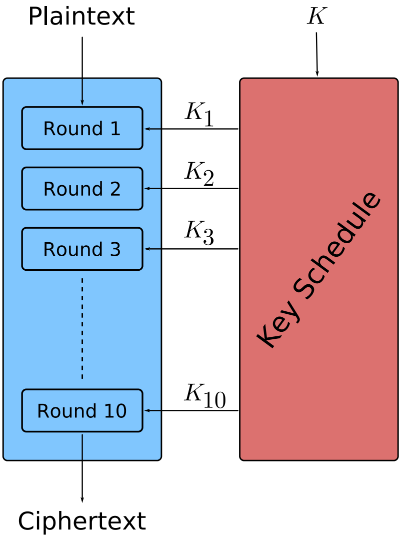
Thomas Baignères   Matthieu Finiasz

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Selected Areas in Cryptography, 2006

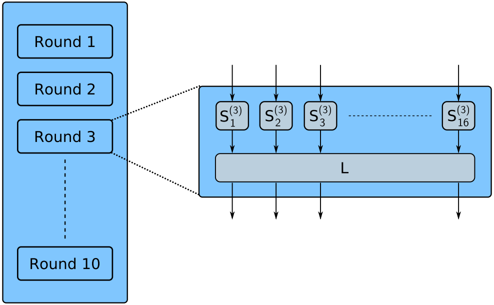
A High Overview of C

- C : $\{0,1\}^{128} \rightarrow \{0,1\}^{128}$ is an iterated block cipher
- $K \in \{0,1\}^{128}$ is the secret key
- Each round i is parameterized by a round key K_i
- $K_1, \ldots, K_{10}$ are derived from K through the key schedule



C is Based on AES's SPN

C is based on a Substitution-Permutation Network (SPN)



Each round is made of:

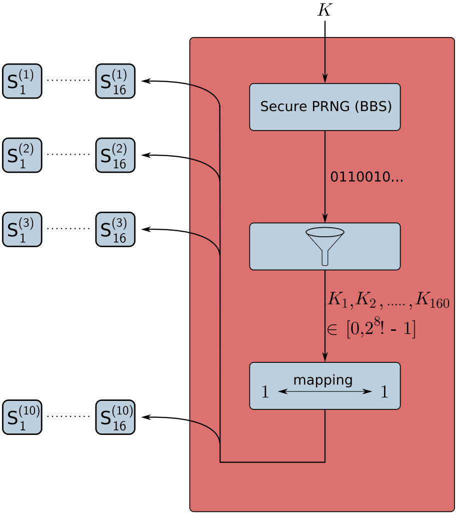
- A layer of substitution boxes $\rightsquigarrow$ confusion
- A linear layer $\rightsquigarrow$ diffusion

- The $\mathsf{S}_i^{(j)}$'s are independent and perfectly random permutations on $\{0,1\}^8$
- The linear layer L is exactly the one used in AES
- Intermediate text values are called a states $\longrightarrow$ elements of $\mathrm{GF}(2^8)^{16}$.

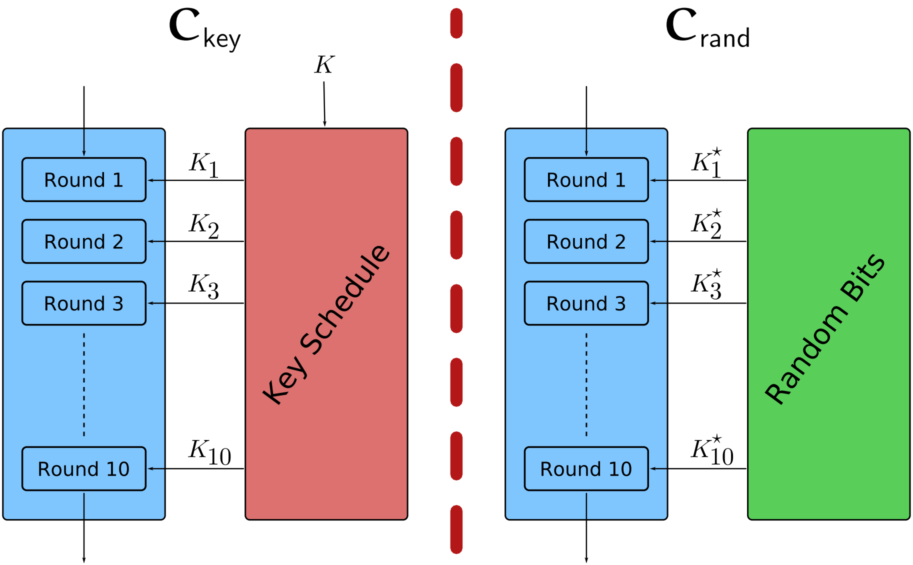
Key Schedule Based on a Cryptographically Secure PRNG

- The Blum-Blum-Shub PRNG generates a long bit string...
- ... from which we extract 160 integers in $[0, 2^8! - 1]$.
- Each of these defines one of the 160 permutations

- The random permutations are computationally indistinguishable from independent and perfectly random permutations.
- We call $K_1, \ldots, K_{160}$ the extended key.
- $\approx 300\ 000$ bits need to be generated.



C_{key} vs. C_{rand}



Previously Known Security Results on $\mathsf{C}_{\mathsf{rand}}$

- Complexity of linear cryptanalysis against $\mathsf{C}_{\mathsf{rand}}$ is on average inversely proportional to
$$\mathrm{ELP}^{\mathsf{C}_{\mathsf{rand}}}(a,b) = \mathrm{E}_{\mathsf{C}_{\mathsf{rand}}}\left((2\Pr_X[a \bullet X = b \bullet \mathsf{C}_{\mathsf{rand}}(X)] - 1)^2\right)$$
- Assuming that all the substitution boxes are independent and perfectly random, Baignères and Vaudenay showed at SAC'05 how to compute the exact value of $\max_{a \neq 0, b} \mathrm{ELP}^{\mathsf{C}_{\mathsf{rand}}}(a,b)$:

2 rounds	3 rounds	4 rounds	6 rounds	8 rounds	9 rounds
$2^{-33.98}$	$2^{-55.96}$	$2^{-127.91}$	$2^{-127.99}$	$2^{-128.00}$	$2^{-128.00}$

$\mathsf{C}_{\mathsf{rand}}$ behaves like the perfect cipher w.r.t. LC and DC when $r \longrightarrow \infty$

Denoting by C^* the perfect cipher, for all non-zero $a, b \in \{0,1\}^{128}$

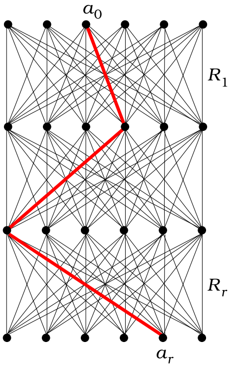
$$\mathrm{ELP}^{\mathsf{C}[r]}(a,b) \xrightarrow[r\to\infty]{} \mathrm{ELP}^{\mathsf{C}^*}(a,b) \quad \text{and} \quad \mathrm{EDP}^{\mathsf{C}[r]}(a,b) \xrightarrow[r\to\infty]{} \mathrm{EDP}^{\mathsf{C}^*}(a,b)$$

About the validity of LC and DC's Security Proofs

- Usual Approximation (red single path):

$$\mathrm{ELP}^{\mathsf{C}_{\mathsf{rand}}}(a_0, a_r) \approx \prod_{i=1}^{r} \mathrm{ELP}^{\mathsf{Round}_i}(a_{i-1}, a_i)$$

- Not always accurate. Leads for AES to $\max_{a\neq 0,b} \mathrm{ELP}^{\mathrm{AES}}(a,b) \approx 2^{-300}$ whereas $\max_{a\neq 0,b} \mathrm{ELP}^{\mathsf{C}^*}(a,b) \approx 2^{-128}$
- The approximation is sufficient for an attack, not for a security proof.



- One needs to consider Nyberg's linear hulls (blue multy paths):

$$\mathrm{ELP}^{\mathsf{C}_{\mathsf{rand}}}(a_0, a_r) = \sum_{a_1,\ldots,a_{r-1}} \prod_{i=1}^{r} \mathrm{ELP}^{\mathsf{Round}_i}(a_{i-1}, a_i)$$

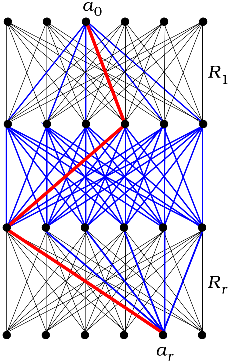
- LC and DC security proofs for $\mathsf{C}_{\mathsf{rand}}$ do take into account linear hulls and differential effects.

About the validity of LC and DC's Security Proofs

- Usual Approximation (red single path):

$$\mathrm{ELP}^{\mathsf{C}_{\mathsf{rand}}}(a_0, a_r) \approx \prod_{i=1}^{r} \mathrm{ELP}^{\mathsf{Round}_i}(a_{i-1}, a_i)$$

- Not always accurate. Leads for AES to $\max_{a\neq 0,b} \mathrm{ELP}^{\mathrm{AES}}(a,b) \approx 2^{-300}$ whereas $\max_{a\neq 0,b} \mathrm{ELP}^{\mathsf{C}^*}(a,b) \approx 2^{-128}$
- The approximation is sufficient for an attack, not for a security proof.



- One needs to consider Nyberg's linear hulls (blue multy paths):

$$\mathrm{ELP}^{\mathsf{C}_{\mathsf{rand}}}(a_0, a_r) = \sum_{a_1,\ldots,a_{r-1}} \prod_{i=1}^{r} \mathrm{ELP}^{\mathsf{Round}_i}(a_{i-1}, a_i)$$

- LC and DC security proofs for $\mathsf{C}_{\mathsf{rand}}$ do take into account linear hulls and differential effects.

From LC to Iterated Attacks of Order 1

- Vaudenay's iterated attacks of order 1 are a generalization of LC.
- In both cases, one bit of information is derived from each text pair.
- LC derives the bit in a linear way.
- No such constraint for Iterated Attacks $\rightsquigarrow$ any kind of binary projection can be used.

Can iterated attack behave any better than LC?

Yes! (see Baignères, Junod, and Vaudenay's Asiacrypt'04 paper).

Provable security of C_{rand} against iterated attacks of order 1

Seven rounds of C_{rand} are sufficient to obtain provable security against any iterated attack of order 1.

Proof (sketch) of the Security of C_{rand} against Iterated Attacks of Order 1

- From the Decorrelation Theory, proving the security against the best non-adaptive 2-limited distinguisher is enough.
- Its advantage is equal to $\frac{1}{2}|||[C_{\mathsf{rand}}]^2 - [C^*]^2|||_\infty$ where

$$[C_{\mathsf{rand}}]^2_{(x_1,x_2),(y_1,y_2)} = \Pr_{C_{\mathsf{rand}}}[C_{\mathsf{rand}}(x_1) = y_1 \,,\, C_{\mathsf{rand}}(x_2) = y_2]$$

- Rounds are mutually independent $\rightsquigarrow [C_{\mathsf{rand}}]^2 = ([\mathsf{Round}]^2)^{10}$
- The trouble is... we have to deal with $2^{256} \times 2^{256}$ matrices!
- Hopefully, the symmetries in the cipher induces symmetries in the matrices.
- Exploiting them leads to computations on 625×625 matrices.

6 rounds	7 rounds	8 rounds	9 rounds	10 rounds	11 rounds
$2^{-71.0}$	$2^{-126.3}$	$2^{-141.3}$	$2^{-163.1}$	$2^{-185.5}$	$2^{-210.8}$

Proof (sketch) of the Security of $\mathsf{C}_{\mathsf{rand}}$ against Iterated Attacks of Order 1

- From the Decorrelation Theory, proving the security against the best non-adaptive 2-limited distinguisher is enough.
- Its advantage is equal to $\frac{1}{2}|||[\mathsf{C}_{\mathsf{rand}}]^2 - [\mathsf{C}^*]^2|||_\infty$ where

$$[\mathsf{C}_{\mathsf{rand}}]^2_{(x_1,x_2),(y_1,y_2)} = \Pr_{\mathsf{C}_{\mathsf{rand}}}[\mathsf{C}_{\mathsf{rand}}(x_1) = y_1 \, , \, \mathsf{C}_{\mathsf{rand}}(x_2) = y_2]$$

- Rounds are mutually independent $\rightsquigarrow [\mathsf{C}_{\mathsf{rand}}]^2 = ([\mathsf{Round}]^2)^{10}$
- The trouble is. . . we have to deal with $2^{256} \times 2^{256}$ matrices!
- Hopefully, the symmetries in the cipher induces symmetries in the matrices.
- Exploiting them leads to computations on 625×625 matrices.

6 rounds	7 rounds	8 rounds	9 rounds	10 rounds	11 rounds
$2^{-71.0}$	$2^{-126.3}$	$2^{-141.3}$	$2^{-163.1}$	$2^{-185.5}$	$2^{-210.8}$

Proof (sketch) of the Security of $\mathsf{C}_{\mathsf{rand}}$ against Iterated Attacks of Order 1

- From the Decorrelation Theory, proving the security against the best non-adaptive 2-limited distinguisher is enough.
- Its advantage is equal to $\frac{1}{2}|||[\mathsf{C}_{\mathsf{rand}}]^2 - [\mathsf{C}^*]^2|||_\infty$ where
$$[\mathsf{C}_{\mathsf{rand}}]^2_{(x_1,x_2),(y_1,y_2)} = \Pr_{\mathsf{C}_{\mathsf{rand}}}[\mathsf{C}_{\mathsf{rand}}(x_1) = y_1\ ,\ \mathsf{C}_{\mathsf{rand}}(x_2) = y_2]$$
- Rounds are mutually independent $\rightsquigarrow [\mathsf{C}_{\mathsf{rand}}]^2 = ([\mathsf{Round}]^2)^{10}$
- The trouble is. . . we have to deal with $2^{256} \times 2^{256}$ matrices!
- Hopefully, the symmetries in the cipher induces symmetries in the matrices.
- Exploiting them leads to computations on 625×625 matrices.

6 rounds	7 rounds	8 rounds	9 rounds	10 rounds	11 rounds
$2^{-71.0}$	$2^{-126.3}$	$2^{-141.3}$	$2^{-163.1}$	$2^{-185.5}$	$2^{-210.8}$

Proof (sketch) of the Security of $\mathsf{C}_{\mathsf{rand}}$ against Iterated Attacks of Order 1

- From the Decorrelation Theory, proving the security against the best non-adaptive 2-limited distinguisher is enough.
- Its advantage is equal to $\frac{1}{2}|||[\mathsf{C}_{\mathsf{rand}}]^2 - [\mathsf{C}^*]^2|||_\infty$ where

$$[\mathsf{C}_{\mathsf{rand}}]^2_{(x_1,x_2),(y_1,y_2)} = \Pr_{\mathsf{C}_{\mathsf{rand}}}[\mathsf{C}_{\mathsf{rand}}(x_1) = y_1\,,\ \mathsf{C}_{\mathsf{rand}}(x_2) = y_2]$$

- Rounds are mutually independent $\rightsquigarrow [\mathsf{C}_{\mathsf{rand}}]^2 = ([\mathsf{Round}]^2)^{10}$
- The trouble is... we have to deal with $2^{256} \times 2^{256}$ matrices!
- Hopefully, the symmetries in the cipher induces symmetries in the matrices.
- Exploiting them leads to computations on 625×625 matrices.

6 rounds	7 rounds	8 rounds	9 rounds	10 rounds	11 rounds
$2^{-71.0}$	$2^{-126.3}$	$2^{-141.3}$	$2^{-163.1}$	$2^{-185.5}$	$2^{-210.8}$

Proof (sketch) of the Security of $\mathsf{C}_{\mathsf{rand}}$ against Iterated Attacks of Order 1

- From the Decorrelation Theory, proving the security against the best non-adaptive 2-limited distinguisher is enough.
- Its advantage is equal to $\frac{1}{2}|||[\mathsf{C}_{\mathsf{rand}}]^2 - [\mathsf{C}^*]^2|||_\infty$ where
$$[\mathsf{C}_{\mathsf{rand}}]^2_{(x_1,x_2),(y_1,y_2)} = \Pr_{\mathsf{C}_{\mathsf{rand}}}[\mathsf{C}_{\mathsf{rand}}(x_1) = y_1 \,,\, \mathsf{C}_{\mathsf{rand}}(x_2) = y_2]$$
- Rounds are mutually independent $\rightsquigarrow [\mathsf{C}_{\mathsf{rand}}]^2 = ([\mathsf{Round}]^2)^{10}$
- The trouble is. . . we have to deal with $2^{256} \times 2^{256}$ matrices!
- Hopefully, the symmetries in the cipher induces symmetries in the matrices.
- Exploiting them leads to computations on 625×625 matrices.

6 rounds	7 rounds	8 rounds	9 rounds	10 rounds	11 rounds
$2^{-71.0}$	$2^{-126.3}$	$2^{-141.3}$	$2^{-163.1}$	$2^{-185.5}$	$2^{-210.8}$

Security of C_{rand} against Impossible Differentials

Definition

A pair of states $a, b \in \mathrm{GF}(2^8)^{16} \setminus \{0\}$ is said to be an impossible differential for C_{rand} if for *any* plaintext x and *any* instance c of C_{rand} we have $\mathsf{c}(x) \oplus \mathsf{c}(x \oplus a) \neq b$.

In other words: an input difference equal to a never leads to an output difference equal to b.

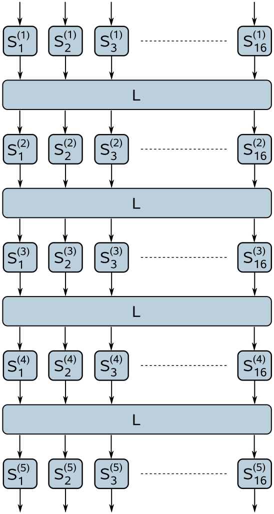
Provable security of C_{rand} against impossible differentials

Given any non-zero input/output differences a and b, there exists at least one instance c of a five-round version of C_{rand} such that

$$\mathsf{c}(0) = 0 \quad \text{and} \quad \mathsf{c}(a) = b.$$

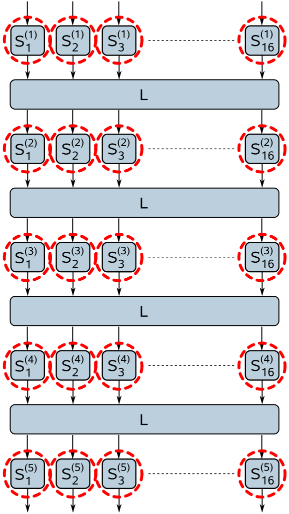
Proof (sketch)

- Defining the instance c means defining the $16 \times 5 = 80$ S-boxes (the only constraint being that the $s_i^{(j)}$'s must be permutations).
- We restrict to permutations s.t. $0 \to 0$, so that $c(0) = 0$.
- Using properties of L $\rightsquigarrow$ first 2 rounds are sufficient to map a on a state of full support, i.e., $\in (\mathrm{GF}(2^8) \setminus \{0\})^{16}$.
- Using the same result backwards, b can be the image of some state of full support.
- The middle S-box layer allows to link both states of full support.
- ...all of this, being consistent with the fact that the $s_i^{(j)}$'s are permutations.



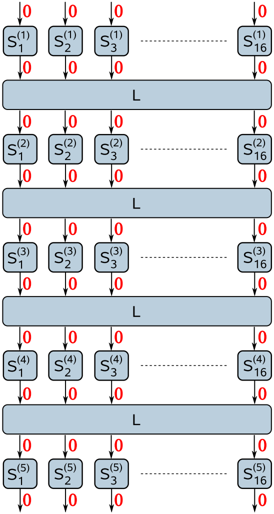
Proof (sketch)

- Defining the instance C means defining the $16 \times 5 = 80$ S-boxes (the only constraint being that the $s_i^{(j)}$'s must be permutations).
- We restrict to permutations s.t. $0 \to 0$, so that $C(0) = 0$.
- Using properties of L $\rightsquigarrow$ first 2 rounds are sufficient to map a on a state of full support, i.e., $\in (\mathrm{GF}(2^8) \setminus \{0\})^{16}$.
- Using the same result backwards, b can be the image of some state of full support.
- The middle S-box layer allows to link both states of full support.
- ...all of this, being consistent with the fact that the $s_i^{(j)}$'s are permutations.



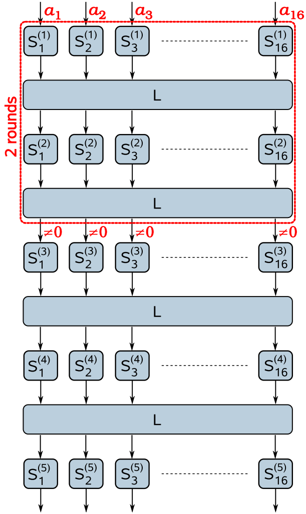
Proof (sketch)

- Defining the instance c means defining the $16 \times 5 = 80$ S-boxes (the only constraint being that the $s_i^{(j)}$'s must be permutations).
- We restrict to permutations s.t. $0 \to 0$, so that $c(0) = 0$.
- Using properties of L $\rightsquigarrow$ first 2 rounds are sufficient to map a on a state of full support, i.e., $\in (\mathrm{GF}(2^8) \setminus \{0\})^{16}$.
- Using the same result backwards, b can be the image of some state of full support.
- The middle S-box layer allows to link both states of full support.
- ...all of this, being consistent with the fact that the $s_i^{(j)}$'s are permutations.



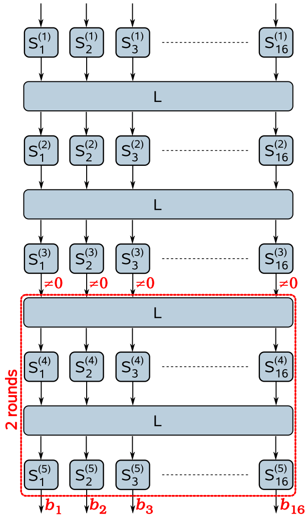
Proof (sketch)

- Defining the instance c means defining the $16 \times 5 = 80$ S-boxes (the only constraint being that the $s_i^{(j)}$'s must be permutations).
- We restrict to permutations s.t. $0 \to 0$, so that $c(0) = 0$.
- Using properties of L $\rightsquigarrow$ first 2 rounds are sufficient to map a on a state of full support, i.e., $\in (\mathrm{GF}(2^8) \setminus \{0\})^{16}$.
- Using the same result backwards, b can be the image of some state of full support.
- The middle S-box layer allows to link both states of full support.
- ...all of this, being consistent with the fact that the $s_i^{(j)}$'s are permutations.



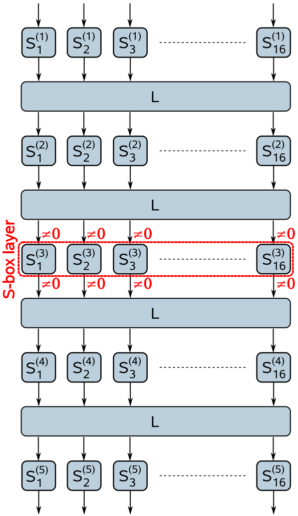
Proof (sketch)

- Defining the instance c means defining the $16 \times 5 = 80$ S-boxes (the only constraint being that the $\mathsf{s}_i^{(j)}$'s must be permutations).
- We restrict to permutations s.t. $0 \to 0$, so that $\mathsf{c}(0) = 0$.
- Using properties of L $\rightsquigarrow$ first 2 rounds are sufficient to map a on a state of full support, i.e., $\in (\mathrm{GF}(2^8) \setminus \{0\})^{16}$.
- Using the same result backwards, b can be the image of some state of full support.
- The middle S-box layer allows to link both states of full support.
- ...all of this, being consistent with the fact that the $\mathsf{s}_i^{(j)}$'s are permutations.



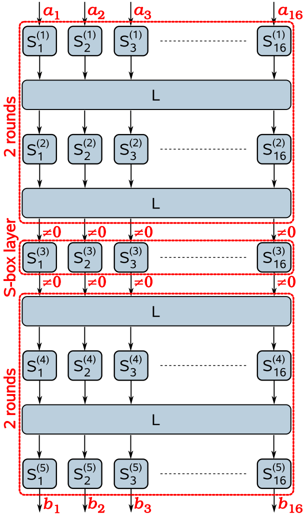
Proof (sketch)

- Defining the instance c means defining the $16 \times 5 = 80$ S-boxes (the only constraint being that the $\mathsf{s}_i^{(j)}$'s must be permutations).
- We restrict to permutations s.t. $0 \to 0$, so that $\mathsf{c}(0) = 0$.
- Using properties of L $\rightsquigarrow$ first 2 rounds are sufficient to map a on a state of full support, i.e., $\in (\mathrm{GF}(2^8) \setminus \{0\})^{16}$.
- Using the same result backwards, b can be the image of some state of full support.
- The middle S-box layer allows to link both states of full support.
- ...all of this, being consistent with the fact that the $\mathsf{s}_i^{(j)}$'s are permutations.



Proof (sketch)

- Defining the instance c means defining the $16 \times 5 = 80$ S-boxes (the only constraint being that the $s_i^{(j)}$'s must be permutations).
- We restrict to permutations s.t. $0 \to 0$, so that $c(0) = 0$.
- Using properties of L $\rightsquigarrow$ first 2 rounds are sufficient to map a on a state of full support, i.e., $\in (\mathrm{GF}(2^8) \setminus \{0\})^{16}$.
- Using the same result backwards, b can be the image of some state of full support.
- The middle S-box layer allows to link both states of full support.
- ...all of this, being consistent with the fact that the $s_i^{(j)}$'s are permutations.



Plugging the Key Schedule In

- In all the security results presented so far, it is assumed that the S-boxes are independent and perfectly random (i.e., valid for C_{rand}).
- This assumption is wrong when using a key schedule with a 128 bit key.
- Although this assumption is sometimes at the origin of potential attacks against block ciphers (weak keys, slide attacks, ...), it still seems to be accepted by the block cipher community.
- The fact that the key schedule of C_{key} is based on a cryptographically secure PRNG allows to relax this assumption: This construction is not limited to C and can be used for any block cipher.

Plugging the Key Schedule In

Provable security of C *with* its key schedule

Under the PRNG security assumption, C used with the key schedule (C_{key}) is as secure as C used with independent and perfectly random boxes (C_{rand}).

- Proof Idea: if there exists an attack much more powerful on C_{key} than on C_{rand}, then there exists a powerful distinguisher on the PRNG.
- In other words: Under the assumption that the PRNG is secure, an attack more efficient against C_{key} than against C_{rand} cannot give the adversary a significant advantage.

Other Security Results

- C is resistant to 2-limited adaptive distinguishers: in the case of C, the advantage of the best 2-limited adaptive adversary is equal to the advantage of the best non-adaptive one.
- The keyed C has no equivalent keys, i.e., the 2^{128} keys define 2^{128} distinct permutations.
- C is (not that) resistant to saturation attacks (aka square attacks): Biryukov and Shamir's attack on SASAS works on 3 rounds of C.
- C seems resistant to algebraic attacks as the s-boxes cannot be described by simple algebraic forms.
- C seems resistant to slide attacks as the key schedule is quite strong.
- C seems resistant against the boomerang attack, differential-linear cryptanalysis, and the rectangle attack, as 4 rounds are sufficient to resist LC and DC.

Other Security Results

- C is resistant to 2-limited adaptive distinguishers: in the case of C, the advantage of the best 2-limited adaptive adversary is equal to the advantage of the best non-adaptive one.
- The keyed C has no equivalent keys, i.e., the 2^{128} keys define 2^{128} distinct permutations.
- C is (not that) resistant to saturation attacks (aka square attacks): Biryukov and Shamir's attack on SASAS works on 3 rounds of C.
- C seems resistant to algebraic attacks as the s-boxes cannot be described by simple algebraic forms.
- C seems resistant to slide attacks as the key schedule is quite strong.
- C seems resistant against the boomerang attack, differential-linear cryptanalysis, and the rectangle attack, as 4 rounds are sufficient to resist LC and DC.

Other Security Results

- C is resistant to 2-limited adaptive distinguishers: in the case of C, the advantage of the best 2-limited adaptive adversary is equal to the advantage of the best non-adaptive one.
- The keyed C has no equivalent keys, i.e., the 2^{128} keys define 2^{128} distinct permutations.
- C is (not that) resistant to saturation attacks (aka square attacks): Biryukov and Shamir's attack on SASAS works on 3 rounds of C.
- C seems resistant to algebraic attacks as the s-boxes cannot be described by simple algebraic forms.
- C seems resistant to slide attacks as the key schedule is quite strong.
- C seems resistant against the boomerang attack, differential-linear cryptanalysis, and the rectangle attack, as 4 rounds are sufficient to resist LC and DC.

Other Security Results

- C is resistant to 2-limited adaptive distinguishers: in the case of C, the advantage of the best 2-limited adaptive adversary is equal to the advantage of the best non-adaptive one.
- The keyed C has no equivalent keys, i.e., the 2^{128} keys define 2^{128} distinct permutations.
- C is (not that) resistant to saturation attacks (aka square attacks): Biryukov and Shamir's attack on SASAS works on 3 rounds of C.
- C seems resistant to algebraic attacks as the s-boxes cannot be described by simple algebraic forms.
- C seems resistant to slide attacks as the key schedule is quite strong.
- C seems resistant against the boomerang attack, differential-linear cryptanalysis, and the rectangle attack, as 4 rounds are sufficient to resist LC and DC.

Other Security Results

- C is resistant to 2-limited adaptive distinguishers: in the case of C, the advantage of the best 2-limited adaptive adversary is equal to the advantage of the best non-adaptive one.
- The keyed C has no equivalent keys, i.e., the 2^{128} keys define 2^{128} distinct permutations.
- C is (not that) resistant to saturation attacks (aka square attacks): Biryukov and Shamir's attack on SASAS works on 3 rounds of C.
- C seems resistant to algebraic attacks as the s-boxes cannot be described by simple algebraic forms.
- C seems resistant to slide attacks as the key schedule is quite strong.
- C seems resistant against the boomerang attack, differential-linear cryptanalysis, and the rectangle attack, as 4 rounds are sufficient to resist LC and DC.

Other Security Results

- C is resistant to 2-limited adaptive distinguishers: in the case of C, the advantage of the best 2-limited adaptive adversary is equal to the advantage of the best non-adaptive one.
- The keyed C has no equivalent keys, i.e., the 2^{128} keys define 2^{128} distinct permutations.
- C is (not that) resistant to saturation attacks (aka square attacks): Biryukov and Shamir's attack on SASAS works on 3 rounds of C.
- C seems resistant to algebraic attacks as the s-boxes cannot be described by simple algebraic forms.
- C seems resistant to slide attacks as the key schedule is quite strong.
- C seems resistant against the boomerang attack, differential-linear cryptanalysis, and the rectangle attack, as 4 rounds are sufficient to resist LC and DC.

Implementing C

- The key schedule is the bottleneck of C $\rightsquigarrow$ it takes 2.5s to a 3.0 GHz Pentium IV to generate the 300'000 bits with BBS.
- To improve this, the random substitution boxes can be drawn in a smaller family than the set of all possible permutations of $\{0,1\}^8$.
- Drawing the boxes in $\mathcal{D}_2 = \{X \mapsto A \oplus \frac{B}{X}\,,\ A, B \in \{0,1\}^8, B \neq 0\}$ does the trick.
- The whole key schedule only requires $2\,560$ bits $\rightsquigarrow$ 100 times faster implementations.

Security considerations

All the proven security results presented on C with perfectly random substitution boxes still hold when drawing the boxes in $\mathcal{D}_2$

Implementing C

- Use AES optimizations: one round of C $\rightsquigarrow$ 16 table look-ups, 12 xors.
- C is slower than AES: each round tables are different from each other...
- ... but the 160 kBytes still fit in the cache of a standard CPU.
- Implementation of C in C on a 3.0 GHz Pentium IV: encryption/decryption speed up to 500 Mbits/s.
- Key schedule takes either 2.5s (perfectly random) or 25ms ($\mathcal{D}_2$).

Applications:

- C cannot be used as a compression function in a MD construction (hashing 1 MByte takes more than one day).
- C with the "fast" key schedule is practical for most encryption/decryption applications.
- C with the "slow" key schedule should be used to reach a very high security level or when the time needed by the key schedule is negligible (e.g. for hard disk encryption).

Implementing C

- Use AES optimizations: one round of C $\rightsquigarrow$ 16 table look-ups, 12 xors.
- C is slower than AES: each round tables are different from each other...
- ... but the 160 kBytes still fit in the cache of a standard CPU.
- Implementation of C in C on a 3.0 GHz Pentium IV: encryption/decryption speed up to 500 Mbits/s.
- Key schedule takes either 2.5s (perfectly random) or 25ms ($\mathcal{D}_2$).

Applications:

- C cannot be used as a compression function in a MD construction (hashing 1 MByte takes more than one day).
- C with the "fast" key schedule is practical for most encryption/decryption applications.
- C with the "slow" key schedule should be used to reach a very high security level or when the time needed by the key schedule is negligible (e.g. for hard disk encryption).

Implementing C

- Use AES optimizations: one round of C $\rightsquigarrow$ 16 table look-ups, 12 xors.
- C is slower than AES: each round tables are different from each other...
- ... but the 160 kBytes still fit in the cache of a standard CPU.
- Implementation of C in C on a 3.0 GHz Pentium IV: encryption/decryption speed up to 500 Mbits/s.
- Key schedule takes either 2.5s (perfectly random) or 25ms ($\mathcal{D}_2$).

Applications:

- C cannot be used as a compression function in a MD construction (hashing 1 MByte takes more than one day).
- C with the "fast" key schedule is practical for most encryption/decryption applications.
- C with the "slow" key schedule should be used to reach a very high security level or when the time needed by the key schedule is negligible (e.g. for hard disk encryption).

Conclusion

- C is a new block cipher (possibly with the slowest key schedule ever).
- C is provably secure against a wide variety of attacks.
- Security proofs still hold when C is used with its key-schedule.
- C is not always practical (still, it *is* in certain cases).
- Some proofs are based on Decorrelation techniques: we don't use decorrelation modules, but take benefit from the symmetries in the cipher to deal with objects that are not as huge as they first seem to be.

Other improvements are possible:

- Use a fast provably secure PRNG, e.g., QUAD (don't miss the first talk tomorrow morning about efficient implementations of multivariate quadratic systems...).
- Further security proofs, e.g., against cache-timing attacks, against d-limited adversaries for $d > 2$,...

Conclusion

- C is a new block cipher (possibly with the slowest key schedule ever).
- C is provably secure against a wide variety of attacks.
- Security proofs still hold when C is used with its key-schedule.
- C is not always practical (still, it *is* in certain cases).
- Some proofs are based on Decorrelation techniques: we don't use decorrelation modules, but take benefit from the symmetries in the cipher to deal with objects that are not as huge as they first seem to be.

Other improvements are possible:

- Use a fast provably secure PRNG, e.g., QUAD (don't miss the first talk tomorrow morning about efficient implementations of multivariate quadratic systems...).
- Further security proofs, e.g., against cache-timing attacks, against d-limited adversaries for $d > 2$,...

Conclusion

- C is a new block cipher (possibly with the slowest key schedule ever).
- C is provably secure against a wide variety of attacks.
- Security proofs still hold when C is used with its key-schedule.
- C is not always practical (still, it *is* in certain cases).
- Some proofs are based on Decorrelation techniques: we don't use decorrelation modules, but take benefit from the symmetries in the cipher to deal with objects that are not as huge as they first seem to be.

Other improvements are possible:

- Use a fast provably secure PRNG, e.g., QUAD (don't miss the first talk tomorrow morning about efficient implementations of multivariate quadratic systems...).
- Further security proofs, e.g., against cache-timing attacks, against d-limited adversaries for $d > 2$,...

Conclusion

- C is a new block cipher (possibly with the slowest key schedule ever).
- C is provably secure against a wide variety of attacks.
- Security proofs still hold when C is used with its key-schedule.
- C is not always practical (still, it *is* in certain cases).
- Some proofs are based on Decorrelation techniques: we don't use decorrelation modules, but take benefit from the symmetries in the cipher to deal with objects that are not as huge as they first seem to be.

Other improvements are possible:

- Use a fast provably secure PRNG, e.g., QUAD (don't miss the first talk tomorrow morning about efficient implementations of multivariate quadratic systems...).
- Further security proofs, e.g., against cache-timing attacks, against d-limited adversaries for $d > 2$,...

Conclusion

- C is a new block cipher (possibly with the slowest key schedule ever).
- C is provably secure against a wide variety of attacks.
- Security proofs still hold when C is used with its key-schedule.
- C is not always practical (still, it *is* in certain cases).
- Some proofs are based on Decorrelation techniques: we don't use decorrelation modules, but take benefit from the symmetries in the cipher to deal with objects that are not as huge as they first seem to be.

Other improvements are possible:

- Use a fast provably secure PRNG, e.g., QUAD (don't miss the first talk tomorrow morning about efficient implementations of multivariate quadratic systems...).
- Further security proofs, e.g., against cache-timing attacks, against d-limited adversaries for $d > 2$,...

Conclusion

- C is a new block cipher (possibly with the slowest key schedule ever).
- C is provably secure against a wide variety of attacks.
- Security proofs still hold when C is used with its key-schedule.
- C is not always practical (still, it *is* in certain cases).
- Some proofs are based on Decorrelation techniques: we don't use decorrelation modules, but take benefit from the symmetries in the cipher to deal with objects that are not as huge as they first seem to be.

Other improvements are possible:

- Use a fast provably secure PRNG, e.g., QUAD (don't miss the first talk tomorrow morning about efficient implementations of multivariate quadratic systems...).
- Further security proofs, e.g., against cache-timing attacks, against d-limited adversaries for $d > 2$,...

The End