FROM RESEARCH TO INDUSTRY

cea tech

Agence Nationale de la Recherche

ANR COGITO

Analyse sécuritaire de l'AES polymorphique

Ph. Jaillon & O. Potin

Palaiseau, 3 décembre 2015.

INVENTEURS DU MONDE NUMÉRIQUE

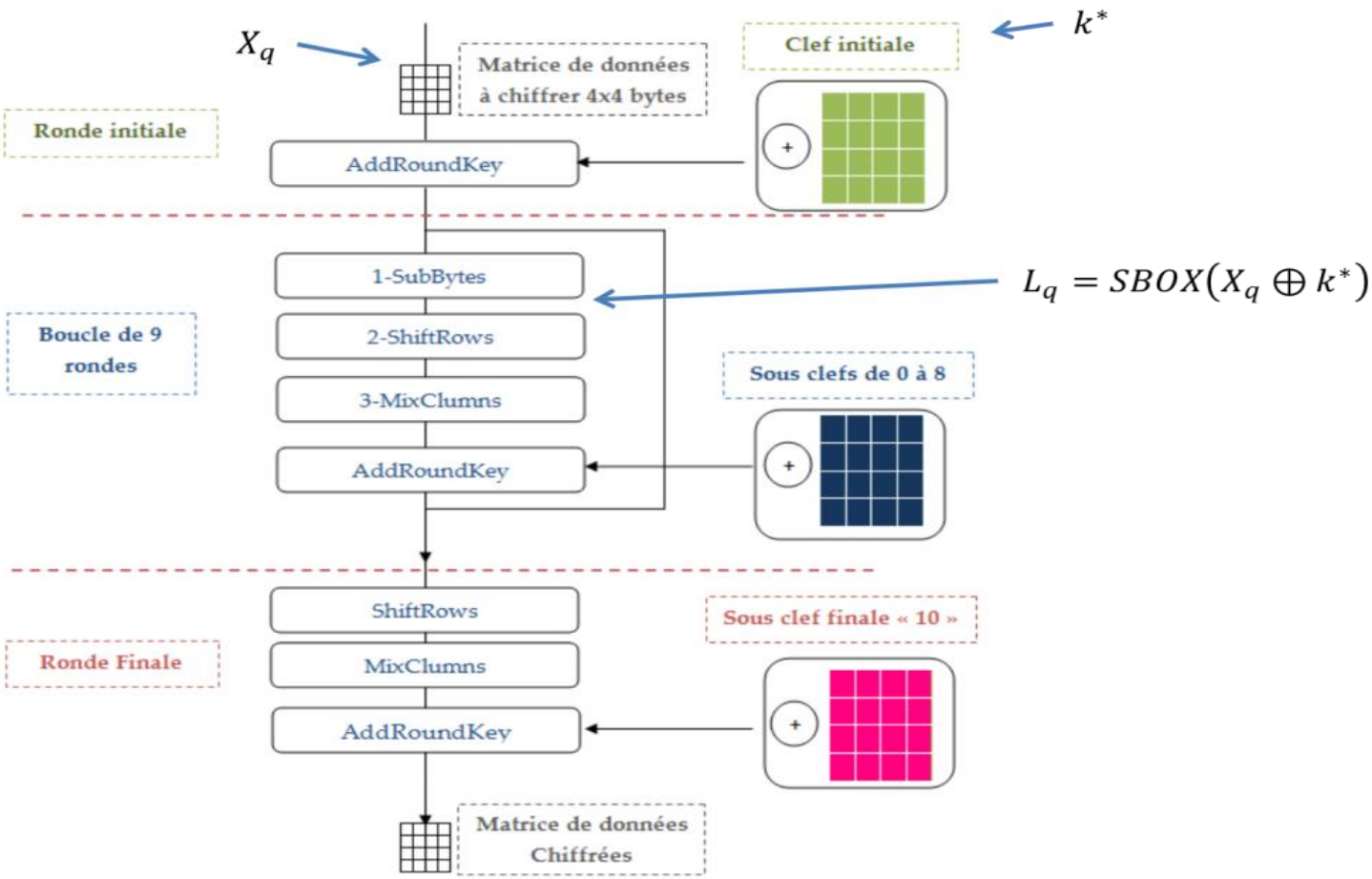
MINES Saint-Étienne

Plan

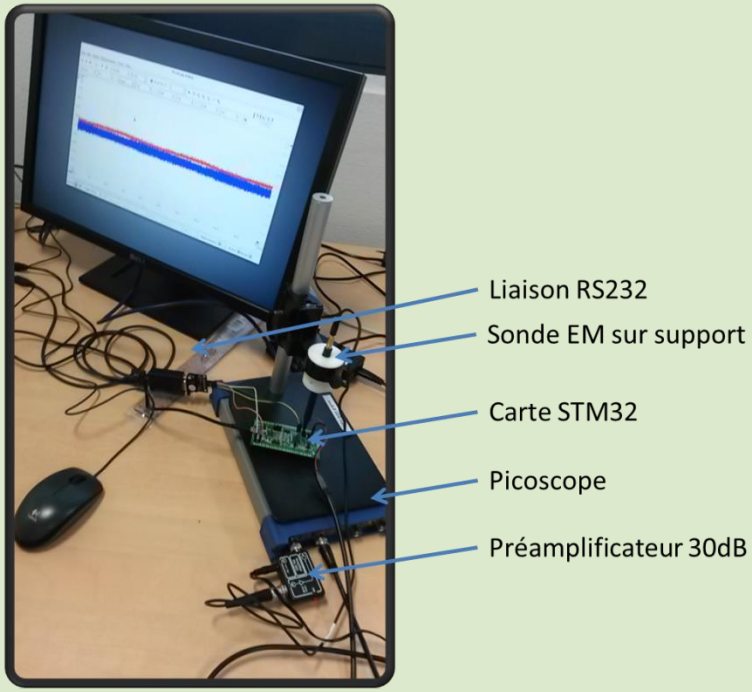
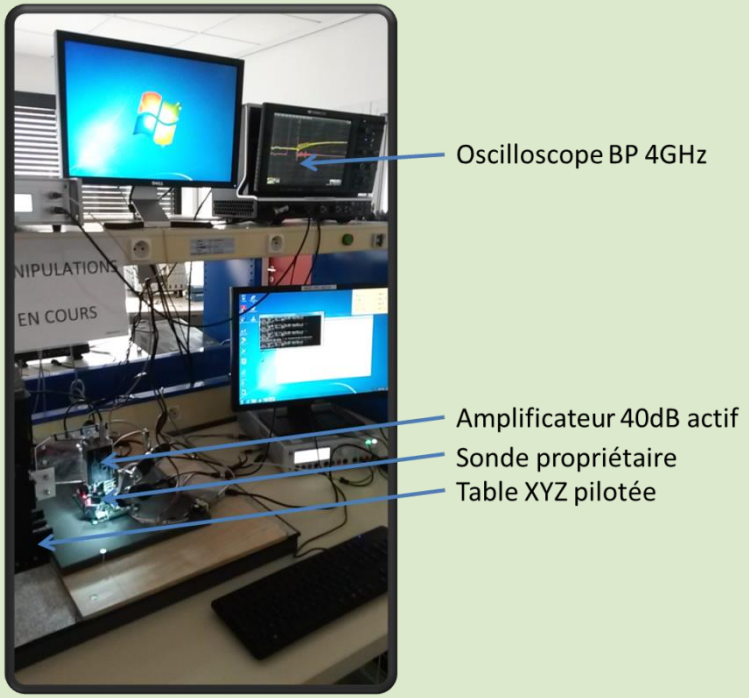
- Contexte
  - AES
- Exploration temporelle
  - Protections COGITO
  - Expérimentations
  - Instruction tracking
- Attaque par canaux cachés
  - CEMA
  - Influence de COGITO
- Perspectives

Use case : AES 8 bits sur STM32 F1

- AES 8 bits
- Fonctions ‘SubBytes’ polymorphiques



Banc de mesures EM

Banc EM faible coût	Banc EM de caractérisation
	
4 bancs mis en œuvre : 2xENSMSE, CEA, INRIA	1 banc mis en œuvre : ENSMSE
Portabilité / Coût / Analyse sécuritaire a minima	Attaque en faute / BP / Profondeur de mémorisation / sonde précise

Exploration temporelle

La génération de code de COGITO modifie les caractéristiques temporelles de l'exécution des programmes.

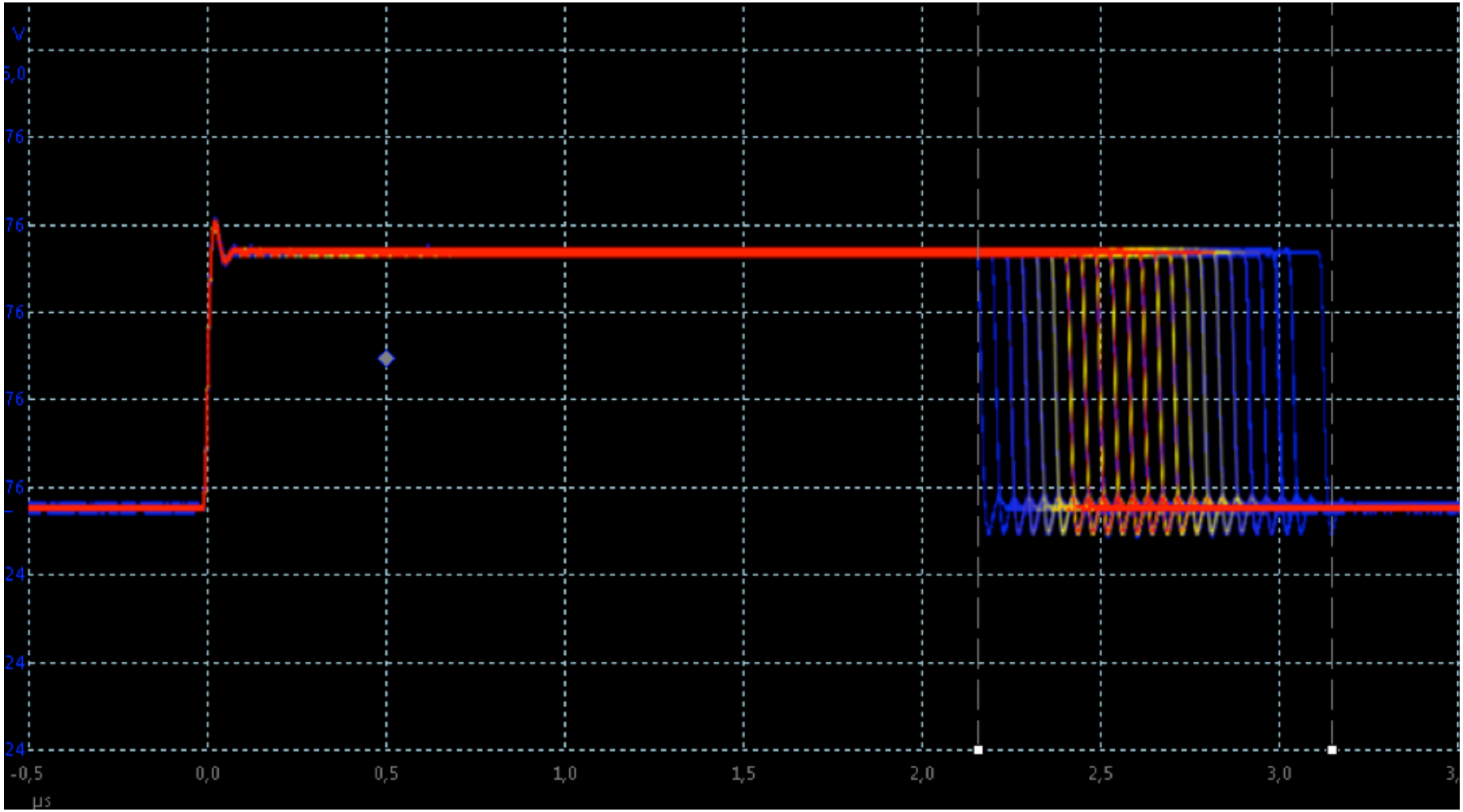
- Etudes dans le cadre du *use case* AES
  - Temps d'exécution de la fonction subBytes()
  - Gigue des instructions

AES: 1er appel à subBytes()

```
void AES_Encrypt(void) {
        addRoundKey();
        for(int i = 0; i < 9; i++) {
                if(i==0) {
                        GPIOC->BSRR = GPIO_Pin_8;
                        subBytes();
                        GPIOC->BRR  = GPIO_Pin_8;
                }
                else subBytes();

                shiftRows();
                mixColumns();
                computeKey(rcon[i]);
                addRoundKey();
        }
        subBytes();
        shiftRows();
        computeKey(rcon[i]);
        addRoundKey();
}
```

Distribution temporelle



Expérimentations

- Pour chaque protection
  - RA, S, IS etN in {0,1}
    et bruit {5, 20, 50, 100}
- Génération de 1000 codes,
  résolution de la durée d'exécution: 40ηs

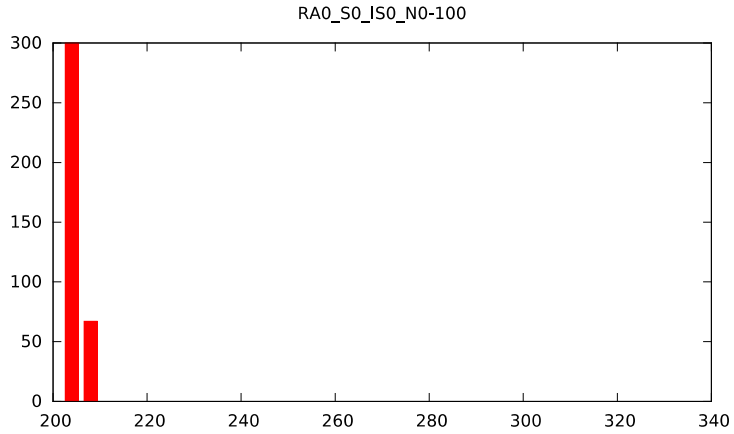
RA Allocation aléatoire de registres

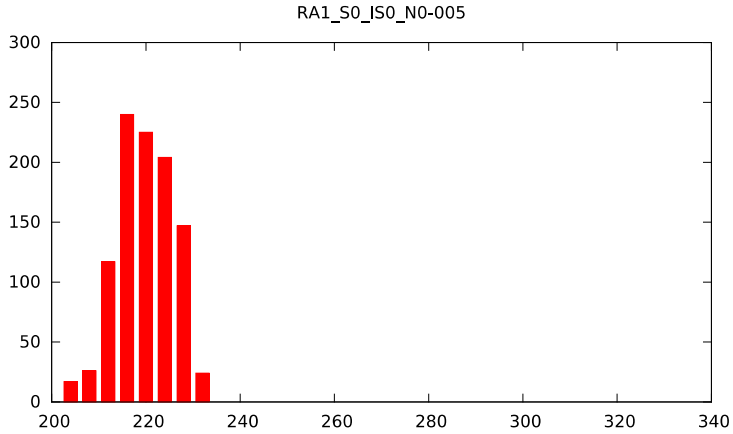
S Substitution d'instructions

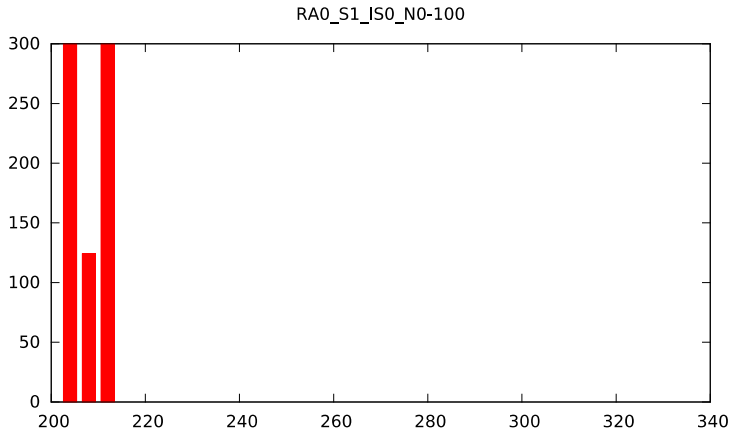
IS Re-ordonnancement d'instructions

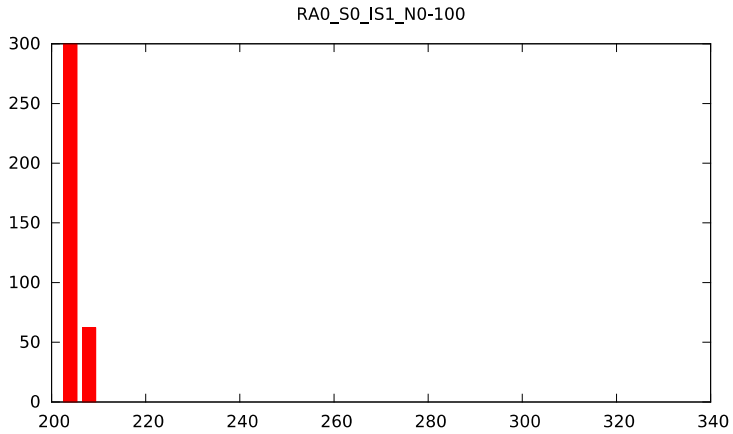
N Insertion de bruit : 5%, 20%, 50%, 100%

Protections

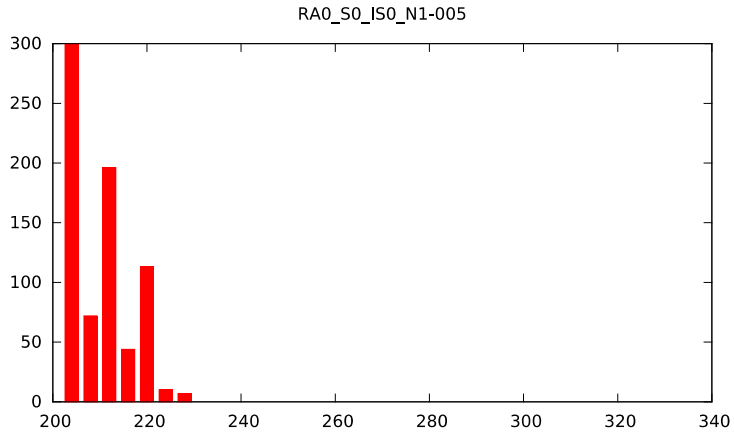


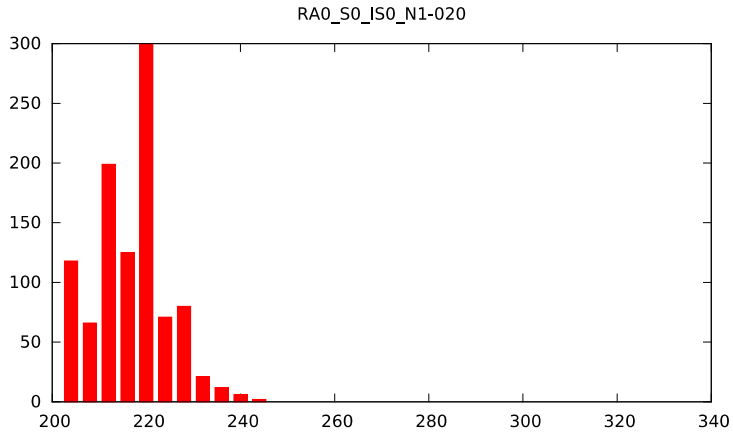


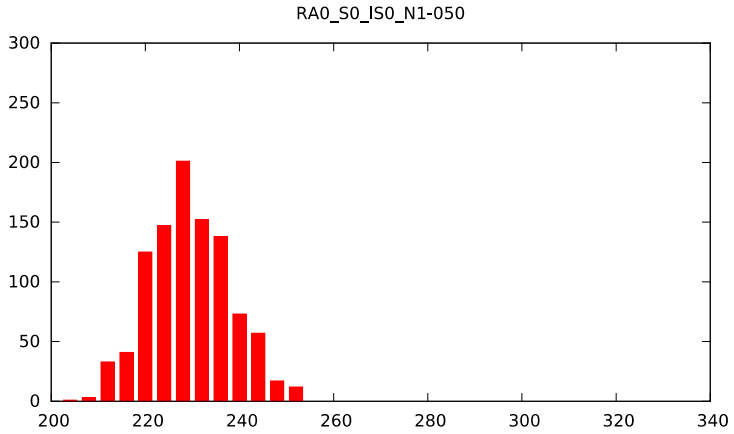


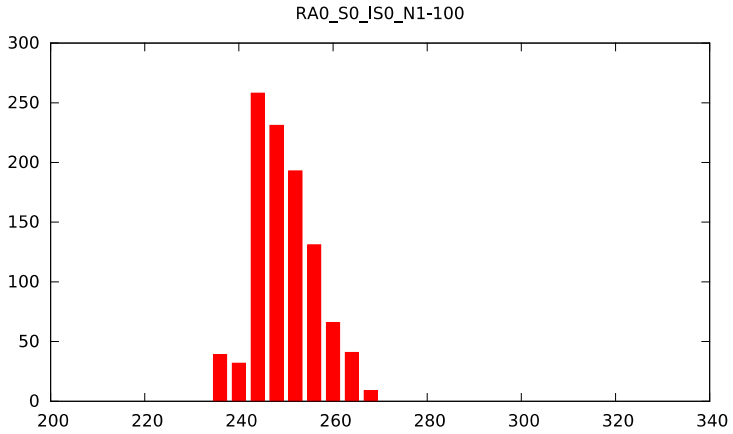


Insertion de bruit

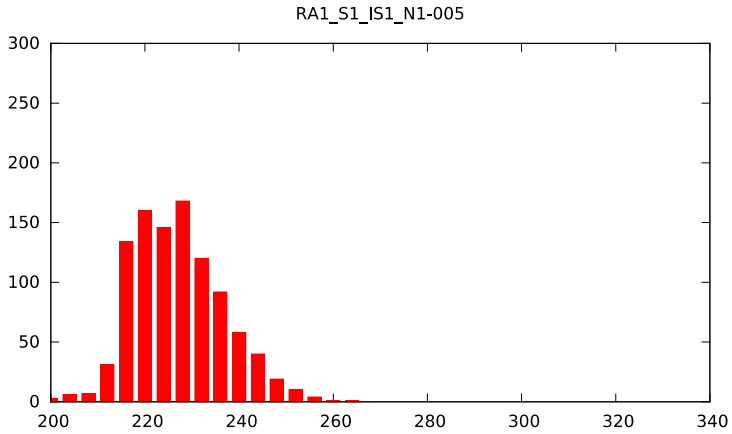


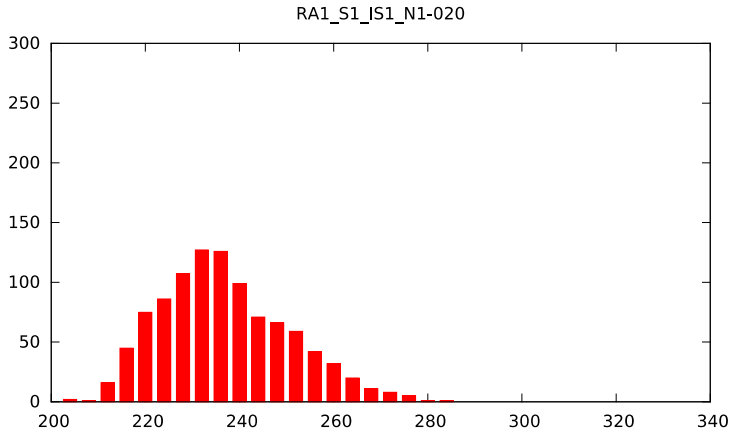


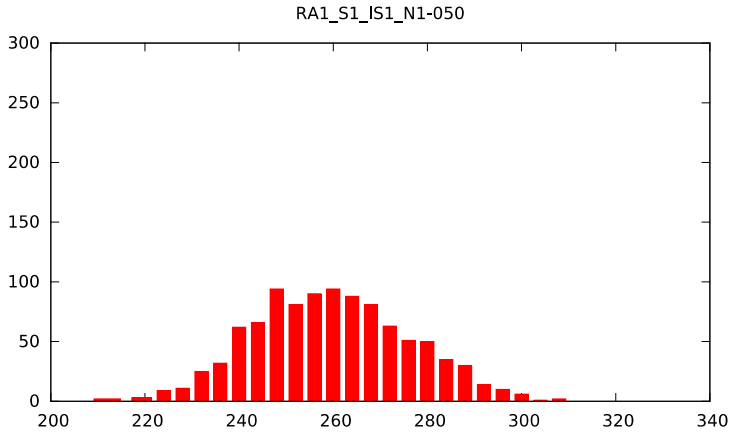


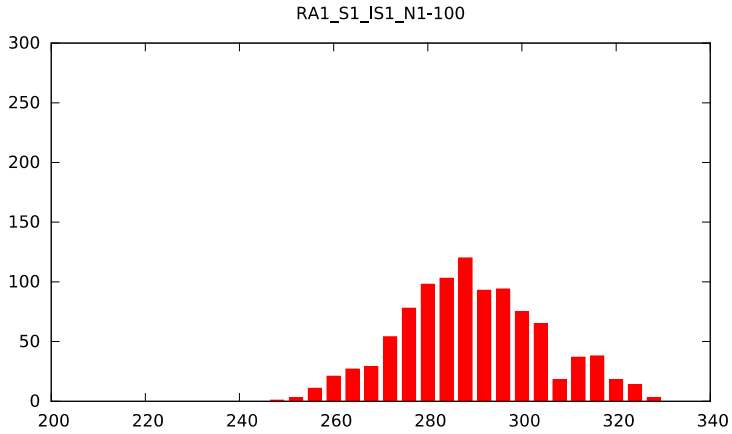


Toutes protections actives









Conclusion

- Les protections COGITO ne masquent pas le signal, mais remplace un *Dirac* par une *Gaussienne*
- Résultats attendus
  - Les attaques nécessitent plus de données pour réussir
  - Importance de la fréquence de mise à jour du code polymorphique.

FROM RESEARCH TO INDUSTRY

cea tech

Agence Nationale de la Recherche

ANR COGITO

INVENTEURS DU MONDE NUMÉRIQUE

MINES Saint-Étienne

Instruction Tracking

Objectif

- Comment tracer une instruction particulière dans le code généré par COGITO ?
- Pas si simple
  - COGITO ajoute des instructions inutiles
  - COGITO re-ordonne les instructions
  - COGITO remplace des instructions
- Idée
  - Utiliser une interruption logicielle à la place de l'instruction tracée et attendre le déroutage dans un "interrupt handler"

Scenario

- gen_subBytes
  (avec instruction SVC – software interrupt)
- Activer le trigger
- Appel à subBytes
  - Exécution de l'instruction SVC
  - Attendre le saut dans le *Interrupt Handler*
    - Inverser le trigger
    - Retourner dans subBytes
  - Continuer l'exécution

Compilette (src code)

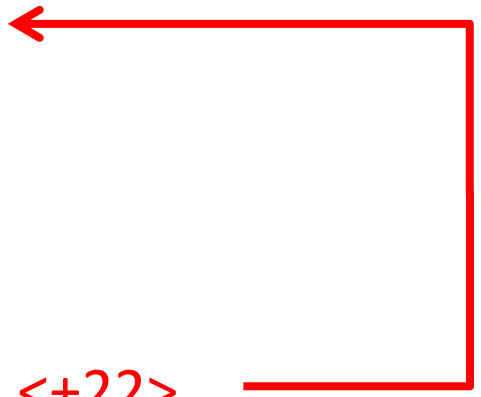
```
void subBytes_compilette(code, sbox_addr, state_addr)
{
#[
    Begin code Prelude
        Type uint32 int 32
        Alloc uint32 rstate
        Alloc uint32 rsbox
        Alloc uint32 rstatei
        Alloc uint32 rsboxi
        Alloc uint32 index

        mv index, #(16)
        loop:
            sub index, index, #(1)
            lb rstatei, rstate, index      //statei = state[i]
            lb rsboxi, rsbox, rstatei      //sboxi = sbox[statei]
            sb rstate, index, rsboxi       //state[i] = sboxi
            bneq loop, index, #(0)

        rtn
    End
]#;
}
```

Compilette (asm code)

```
0x2000004c <>:      stmdb  sp!, {r4, r6, r7, r9, r10, lr}
0x20000050 <+4>:    movw   lr, #60          ; state address
0x20000054 <+8>:    movt   lr, #8192
0x20000058 <+12>:   movw   r9, #5236        ; sbox address
0x2000005c <+16>:   movt   r9, #2048
0x20000060 <+20>:   movs   r4, #16
0x20000062 <+22>:   subs   r4, #1
0x20000064 <+24>:   ldrb.w r10, [lr, r4]
0x20000068 <+28>:   ldrb.w r6, [r9, r10]
0x2000006c <+32>:   strb.w r6, [lr, r4]
0x20000070 <+36>:   cmp    r4, #0
0x20000072 <+22>:   bne.n  0x20000062 <+22>
0x20000074 <+40>:   ldmia.w        sp!, {r4, r6, r7, r9, r10, lr}
0x20000078 <+44>:   bx     lr
```

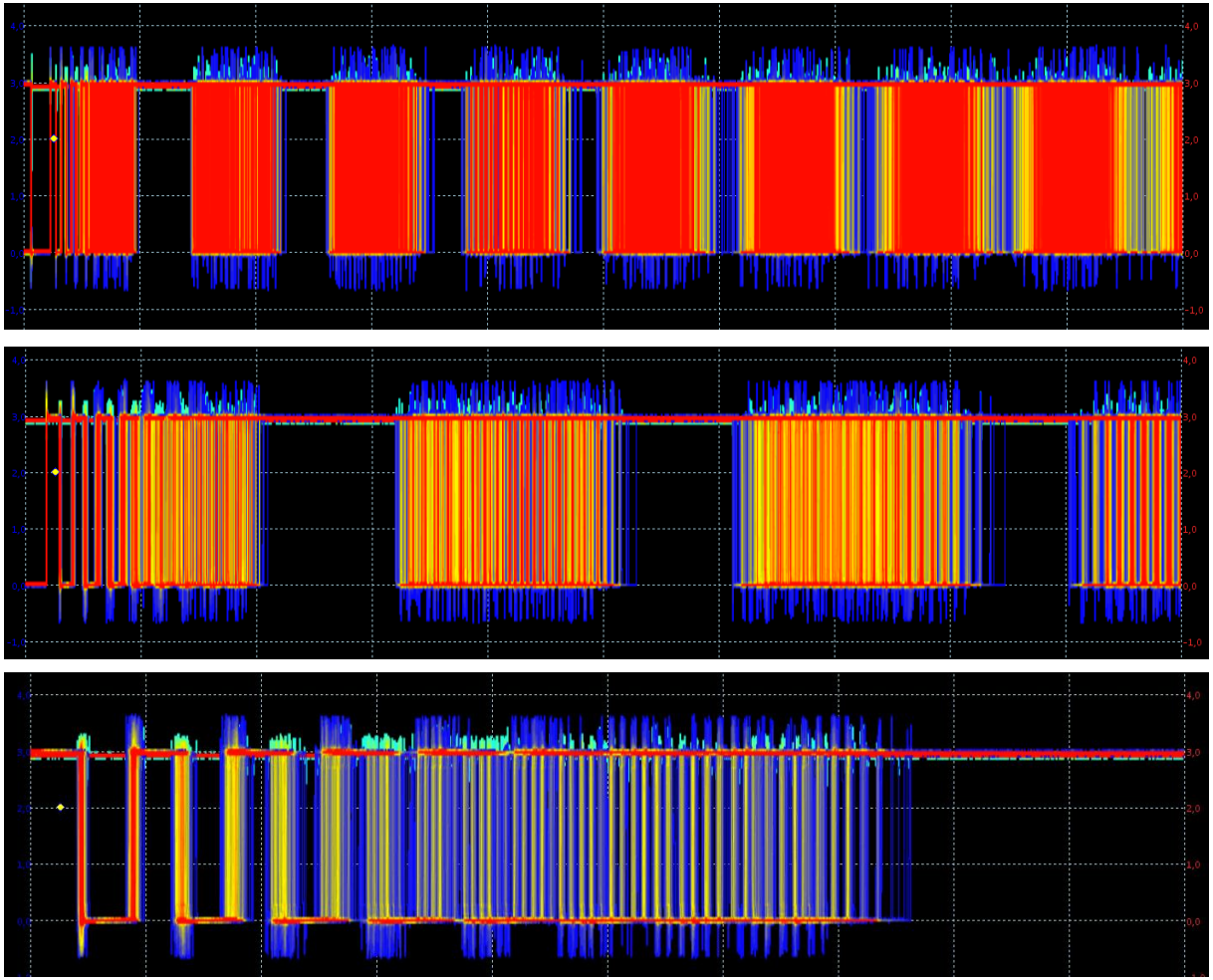


Résultats

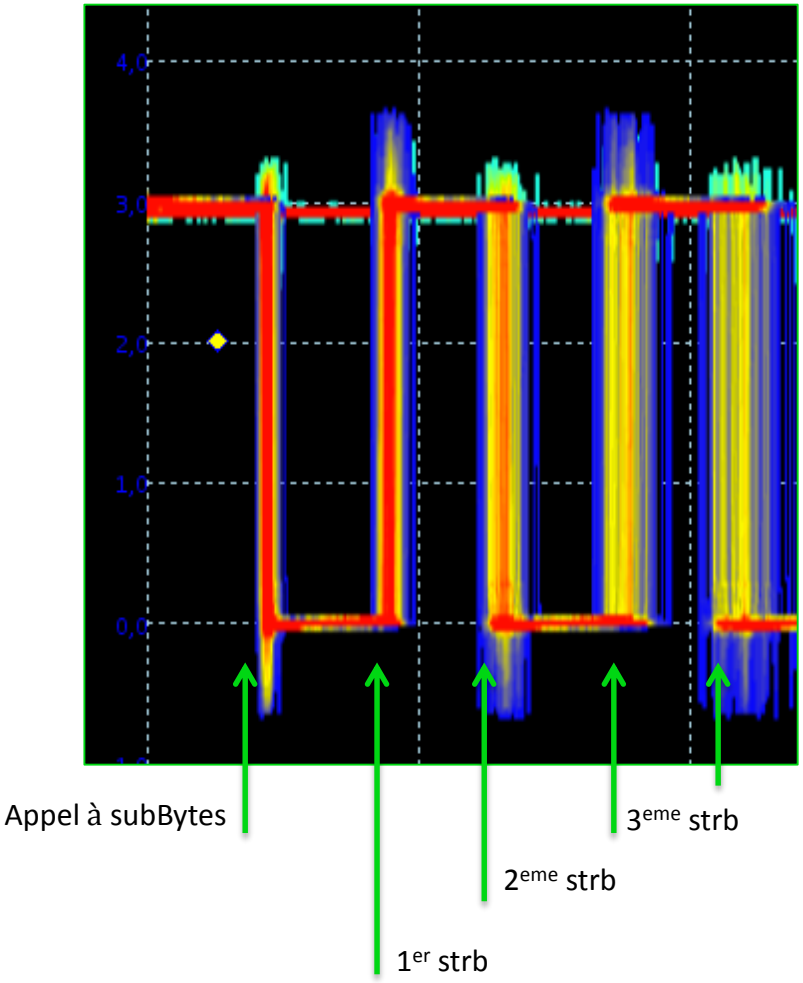
- Toutes les phases d'optimisations de deGoal sont préservées
  - Optimisation et protection Cogito/deGoal sont indépendantes de l'opcode des instructions.
- Bruit et allocation de registres produisent une répartition Gaussienne du moment de l'exécution des l'instructions tracées

- Le programme exécuté n'est pas fonctionnel
- seulement 4 instructions dans subBytes
  - Le re-ordonnancement est limité !

Observations

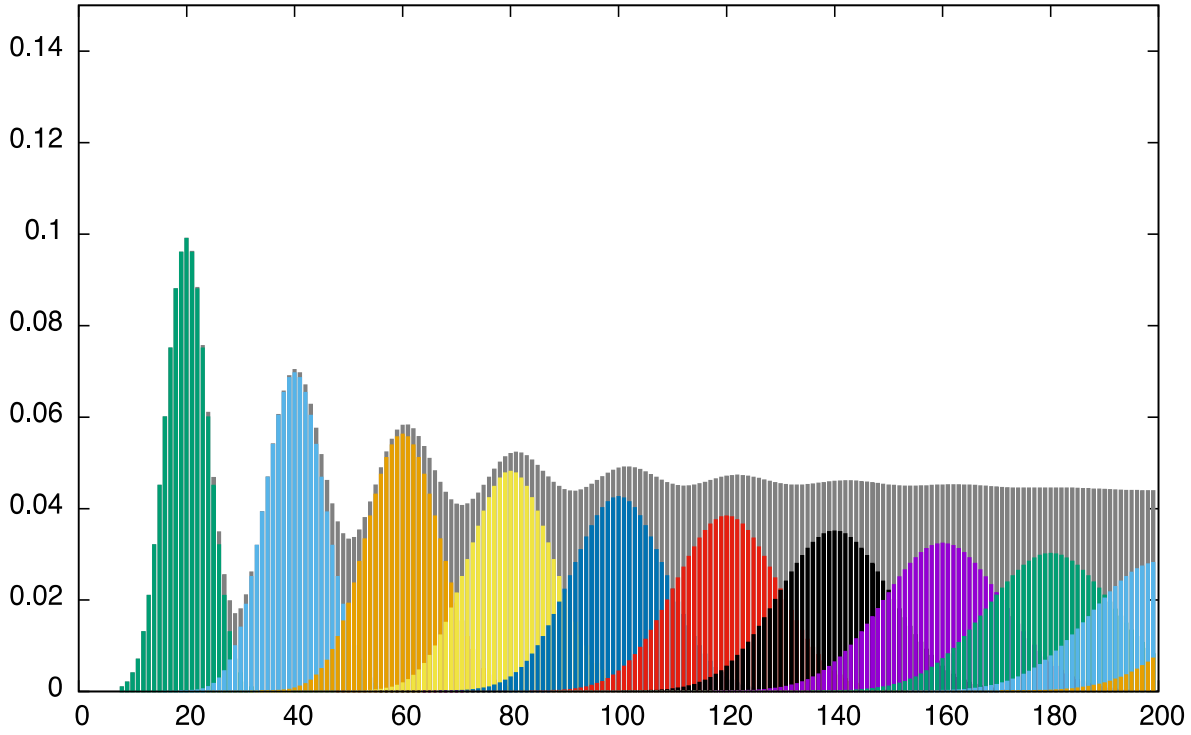


Observations (zoom)



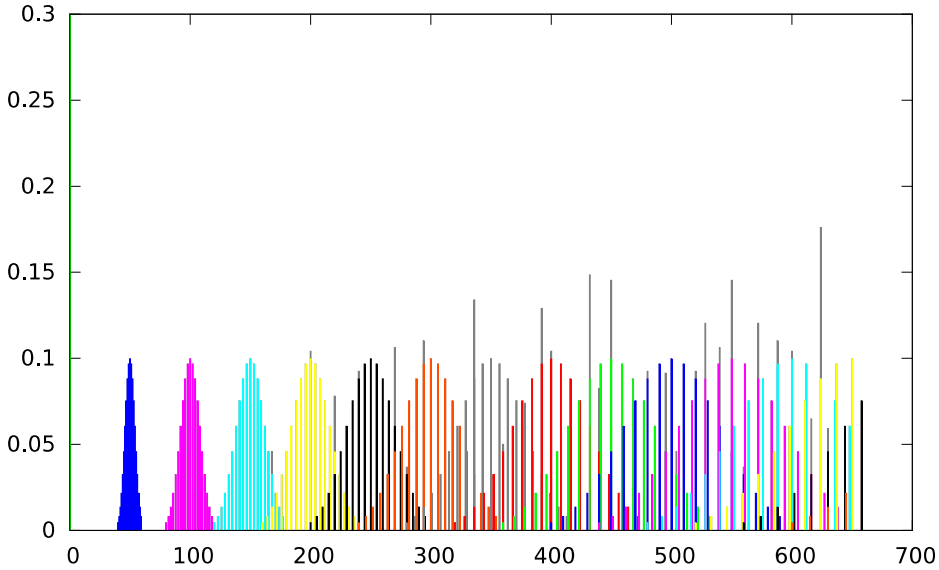
Distribution normale d'instructions

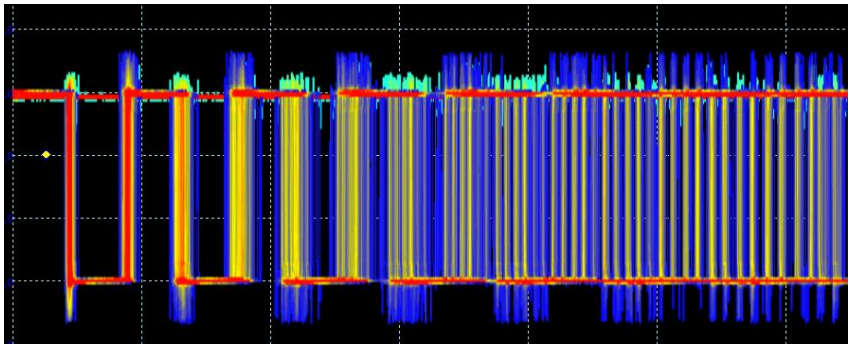
- Convolution de fonctions de densité de probabilité



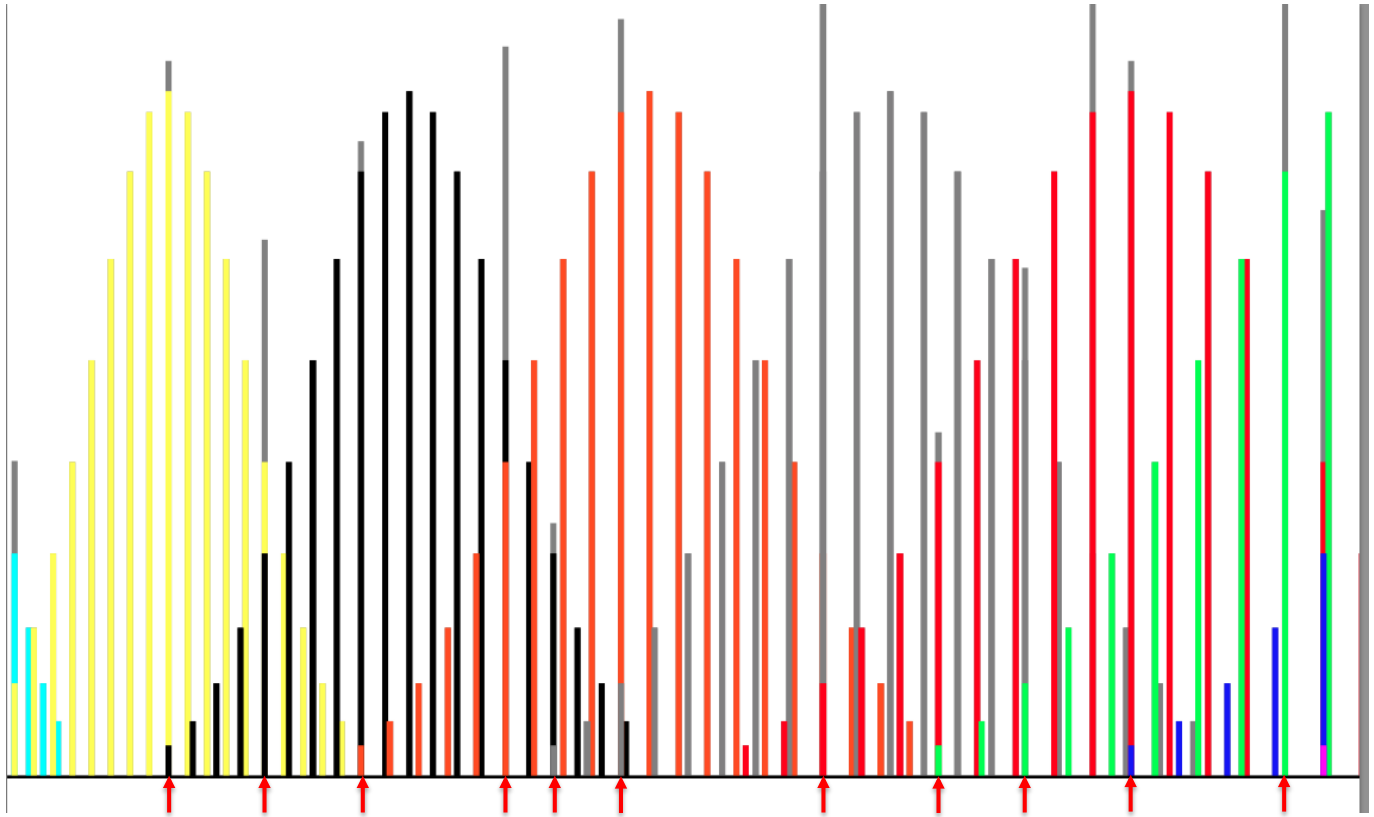

```
for(i=0 ; i<N ; i++) {
        track_instruction(OP);
        wait(To+f(dt)) ;
}
```

```
t = To+f(dt) ;
for(i=0 ; i<N ; i++) {
        track_instruction(OP);
        wait(t) ;
}
```





- Les codes générés respectent une loi normale.
- Absence de fréquences communes entre les différentes classes de code.



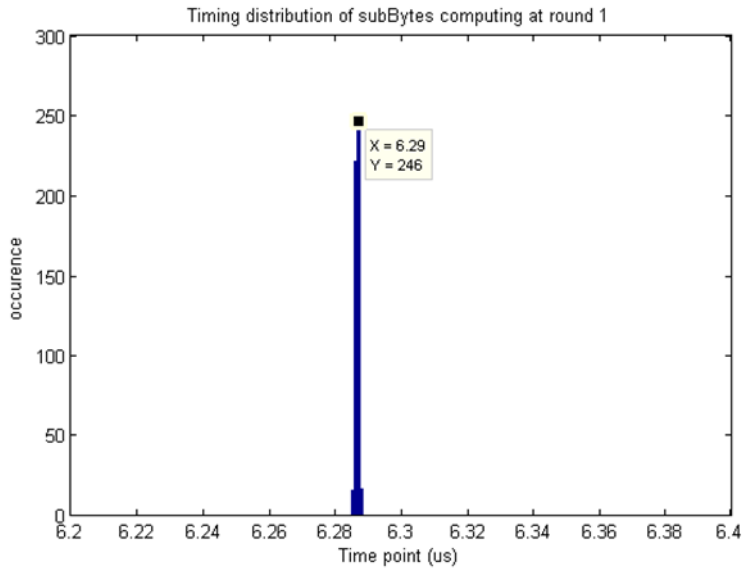
Recouvrements,

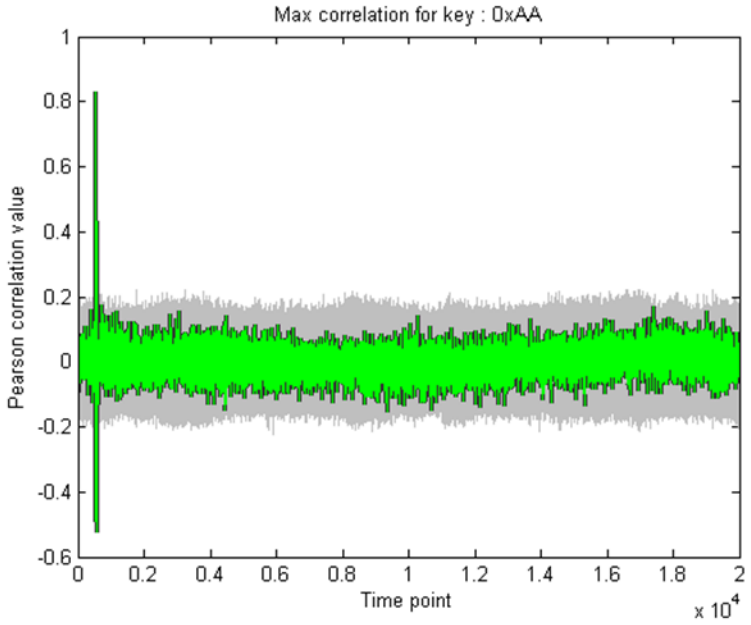
Plan

- Contexte
  - AES
- Exploration temporelle
  - Protections COGITO
  - Expérimentations
  - Instruction tracking
- Attaque par canaux cachés
  - CEMA
  - Influence de COGITO
- Perspectives

CEMA sur AES sans protection

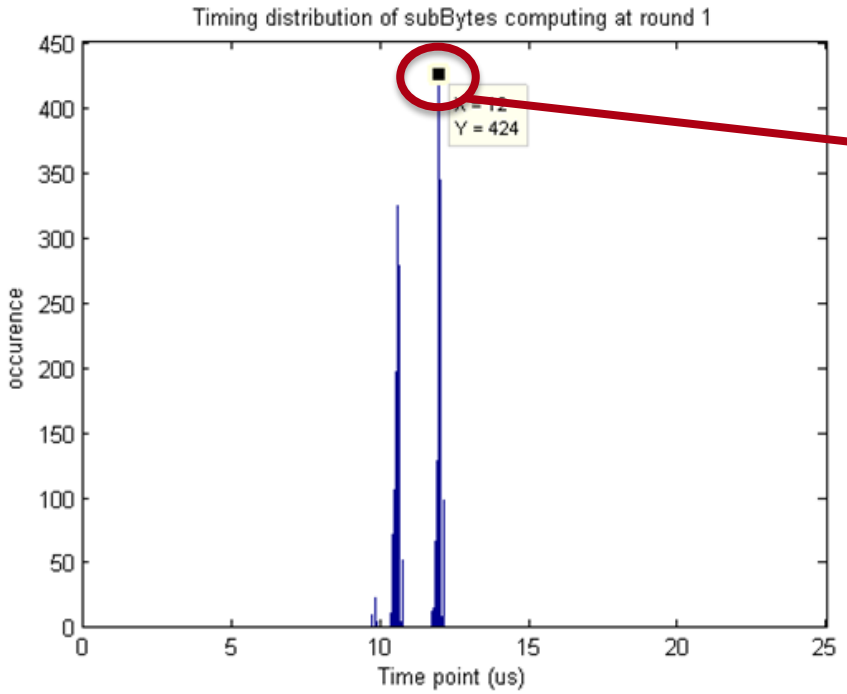
- Modèle de fuite : Poids de Hamming
- 50 courbes pour 60% de ‘success rate’
- Corrélation de Pearson : 0,8 ➔ Key = 0xAA

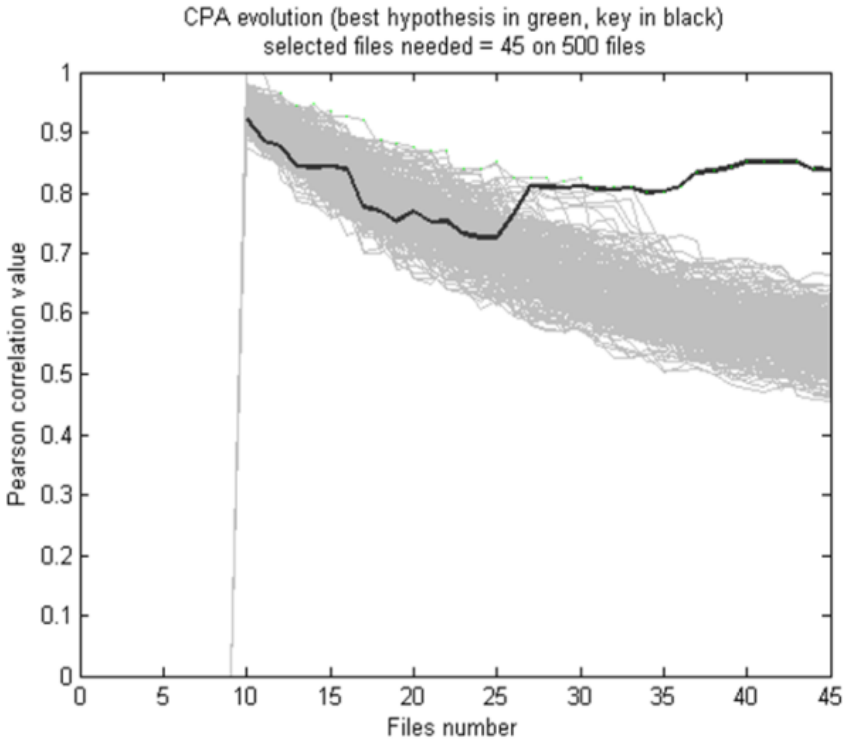




Evaluation par protection : RA

- RA : ‘Register Allocation’

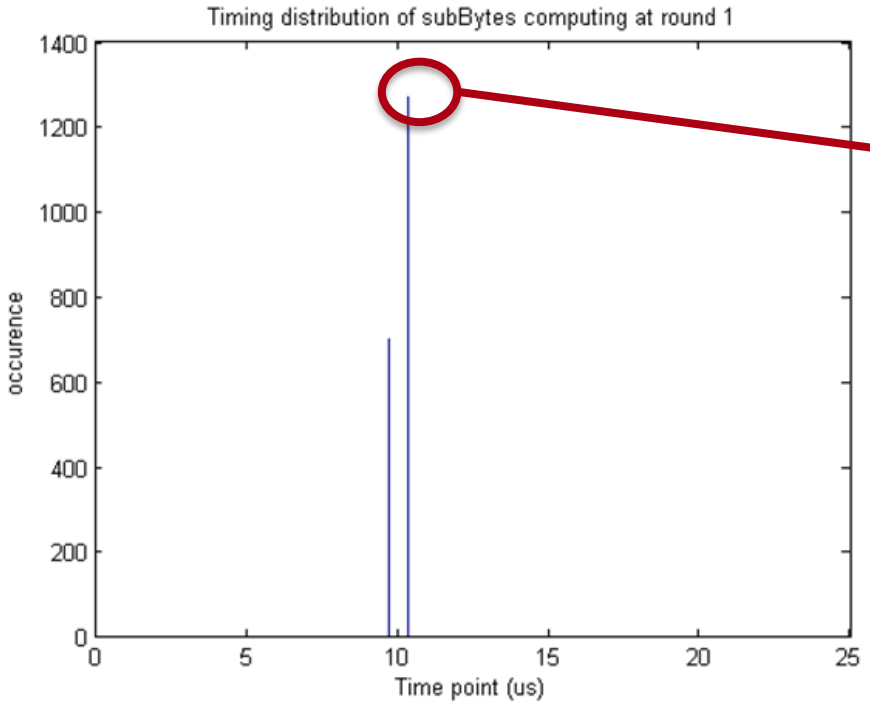


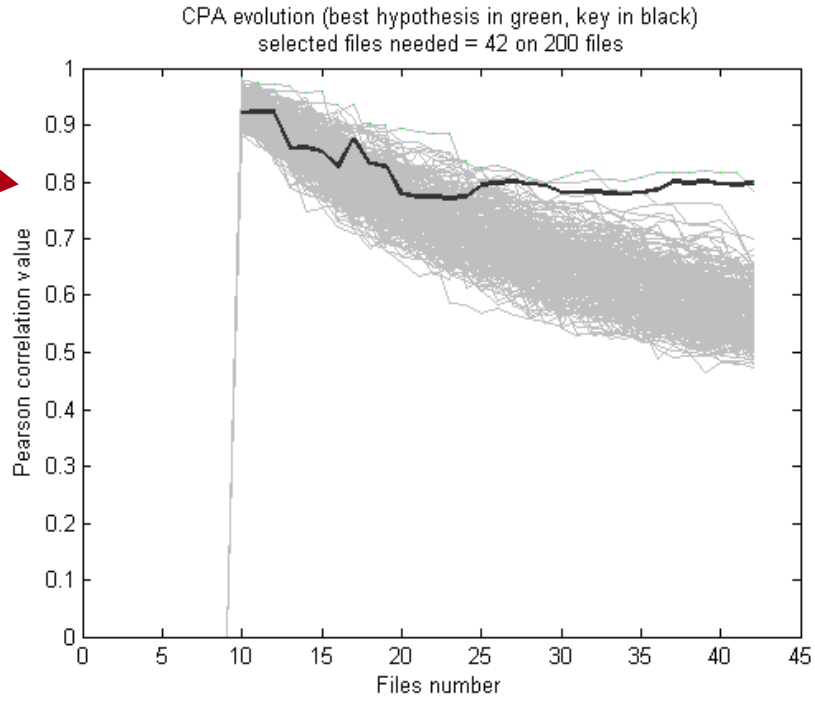


Protection	CPA	# courbes	CPA (regroupement)	# courbes utiles / # courbes total	Facteur de sécurité
RA	0,11	5000	CPA = 0,83	45 / 500	10

Evaluation par protection : IS

- IS : Instruction Substitution

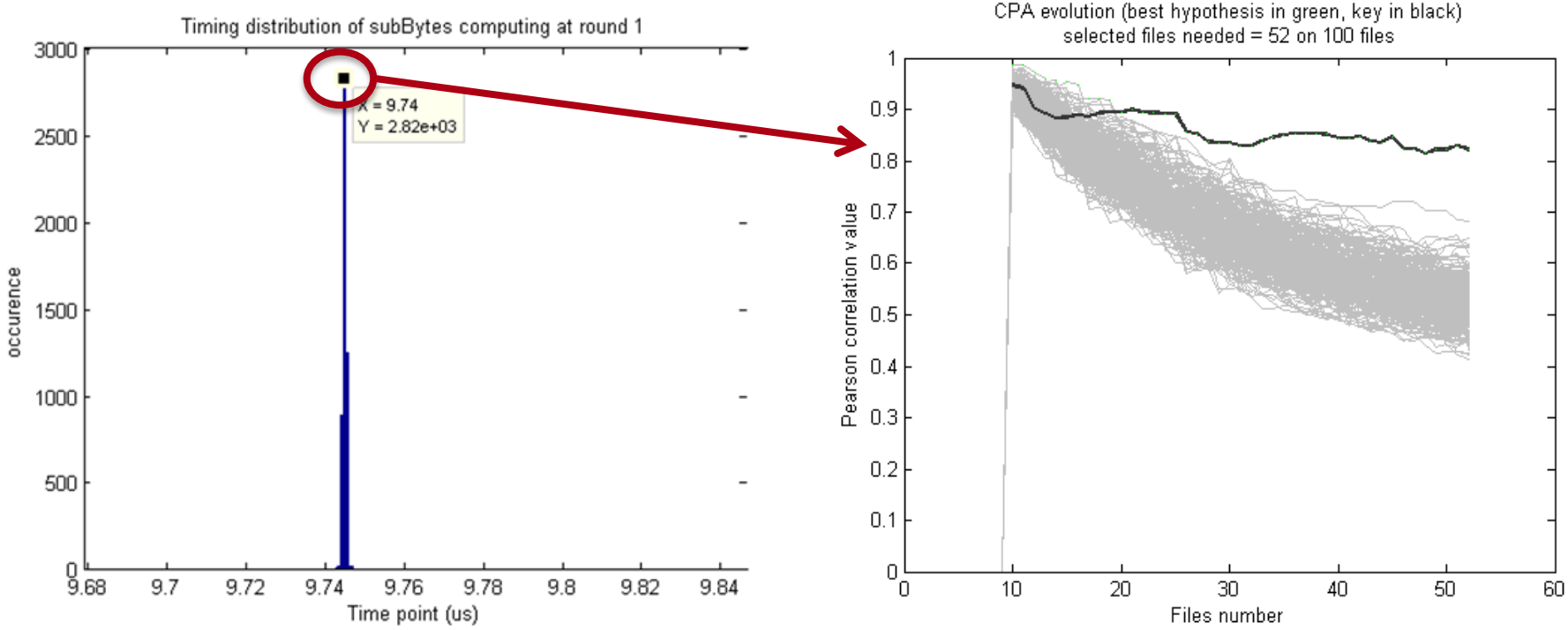




Protection	CPA	# courbes	CPA (regroupement)	# courbes utiles / # courbes total	Facteur de sécurité
IS	0,3	5000	0,8	42 / 200	4

Evaluation par protection : S

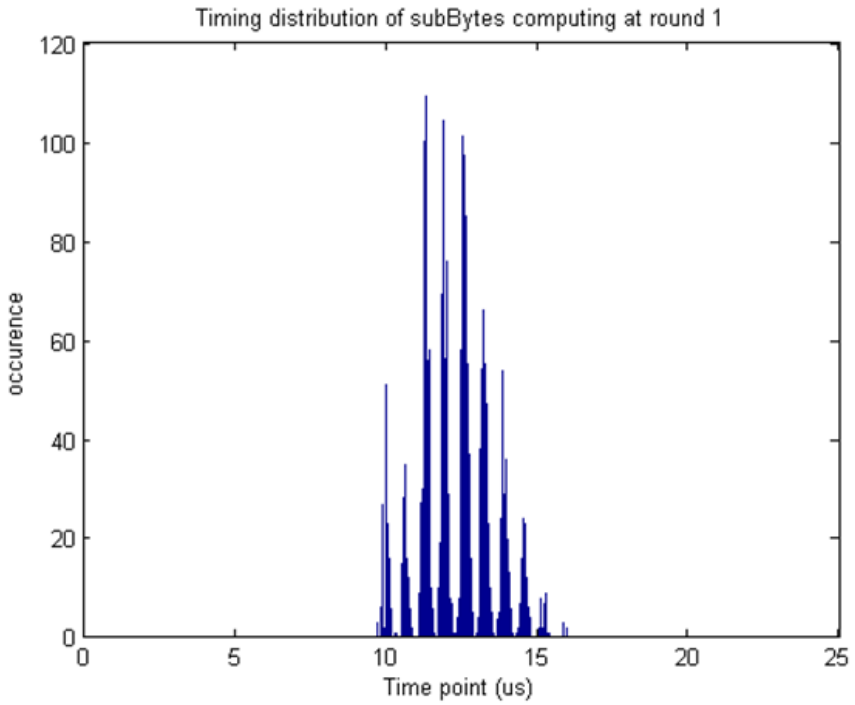
- S : Ordonnancement des instructions (*shuffling*)

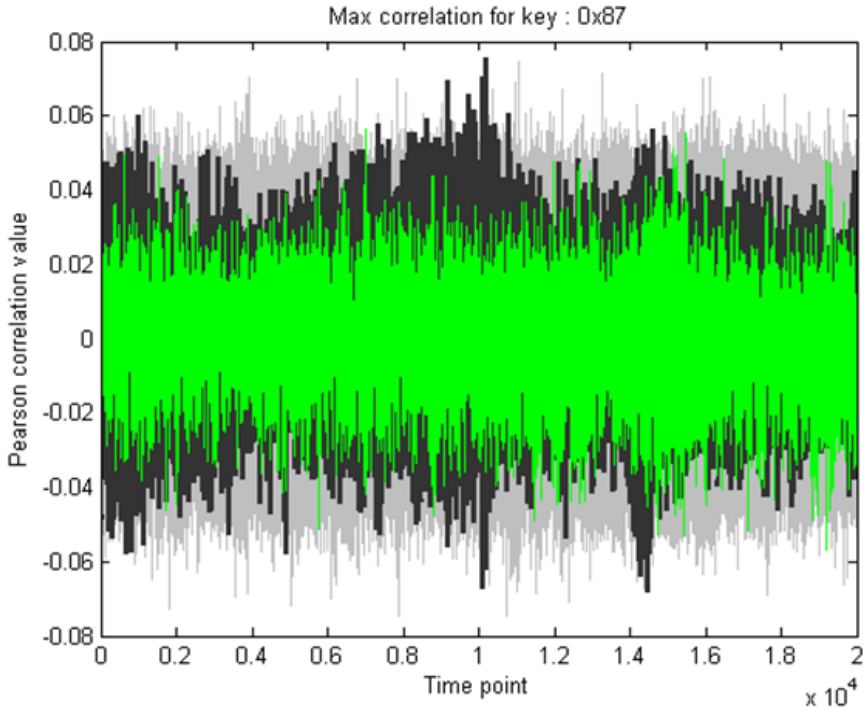


Protection	CPA	# courbes	CPA (regroupement)	# courbes utiles / # courbes total	Facteur de sécurité
IS	0,7	5000	0,8	52 / 100	2

Evaluation par protection : N

- N : Noise





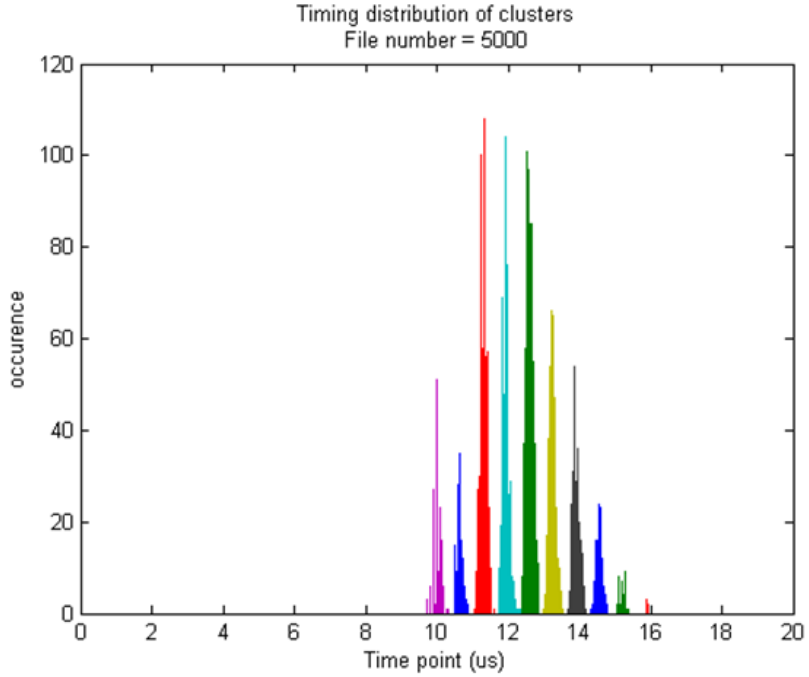
Protection	CPA	# courbes
Noise	0.08	5000

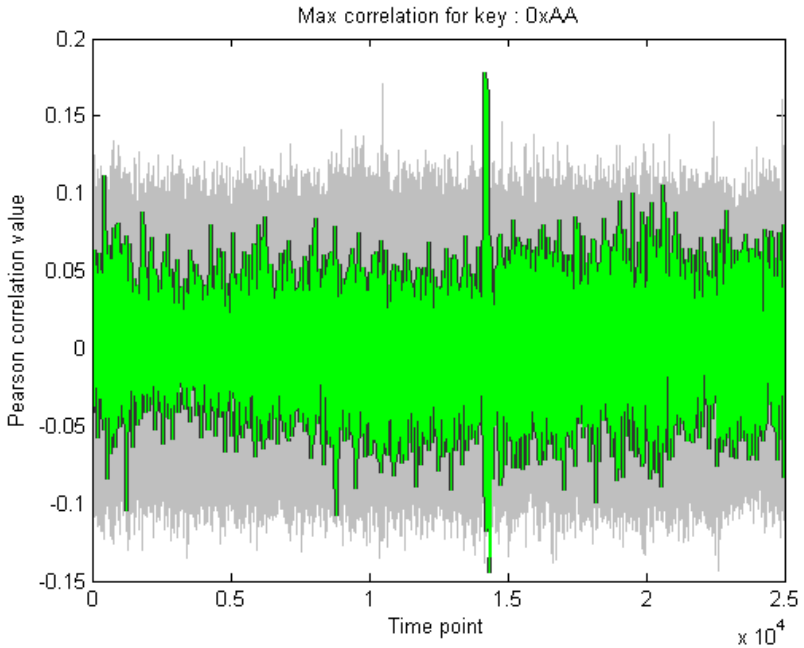
Evaluation par protection : N

- Regroupement par ‘Magnitude Squared Coherence’

$$MSC(f) = \frac{|P_{xy}(f)|^2}{P_{xx}(f).P_{yy}(f)} \quad avec\ P_{xy} = densité\ spectrale\ et\ P_{xx} = spectre\ croisé$$

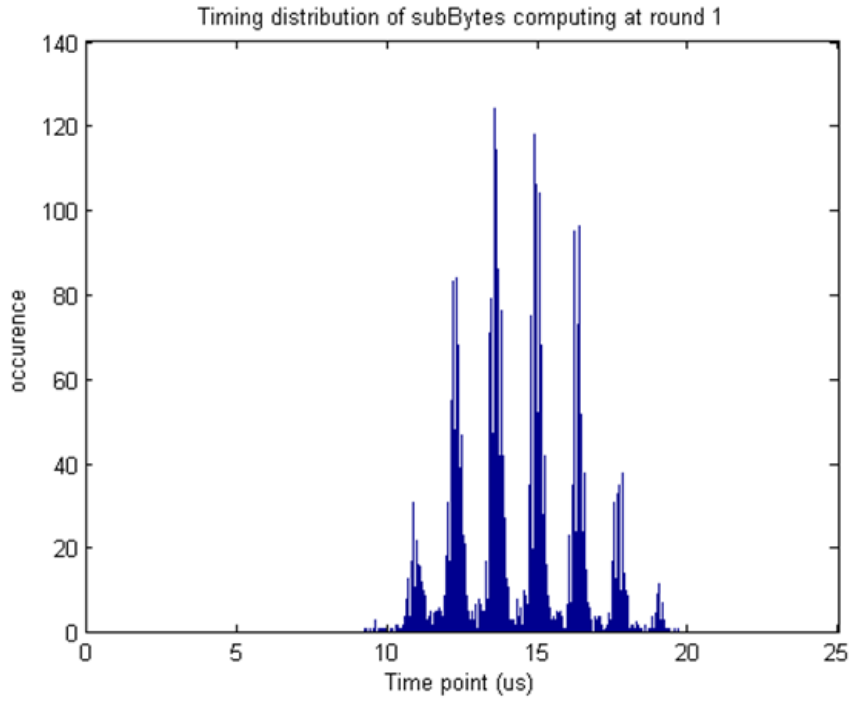
- 10 regroupements / 1114 courbes
  - **Facteur de sécurité = 100 ; CPA = 0.15**

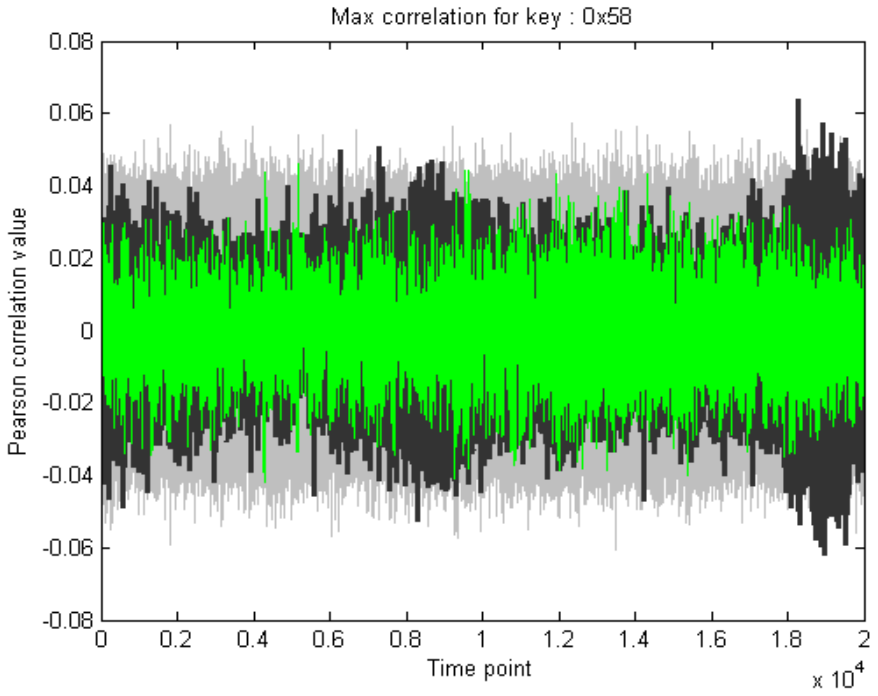




Toutes les protections

- RA + IS + S + N

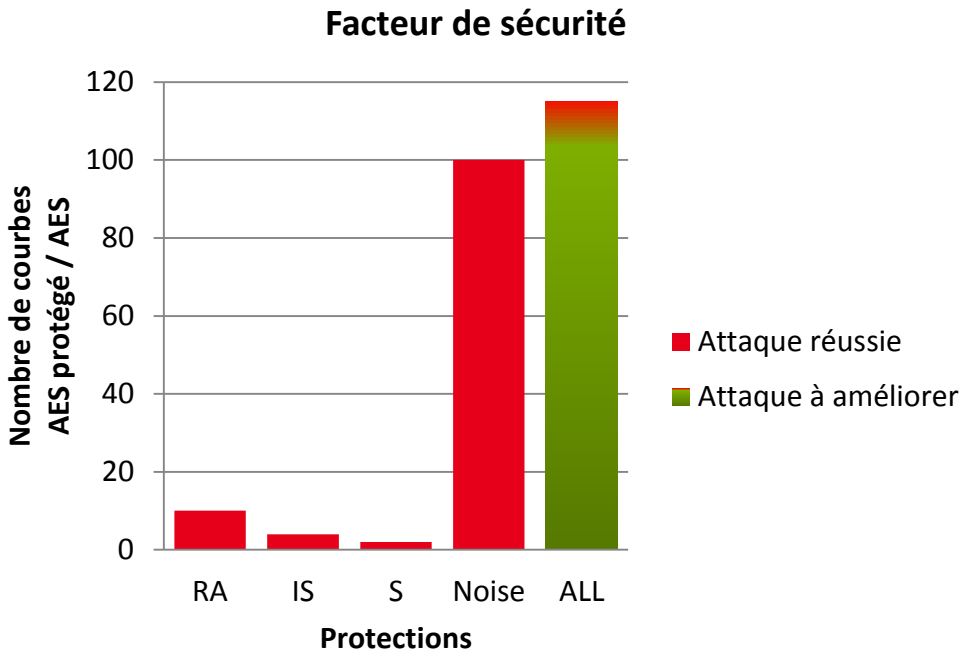




Protection	CPA	# courbes	CPA (regroupement)	# courbes utiles / # courbes total	Facteur de sécurité
IS	0,06	5000	0,13	1251 / 5000	>100

Résumé

- Register Allocation, Instruction Substitution et Shuffling => faible facteur de sécurité
- Noise et protections cumulées
  - facteur de sécurité > 100



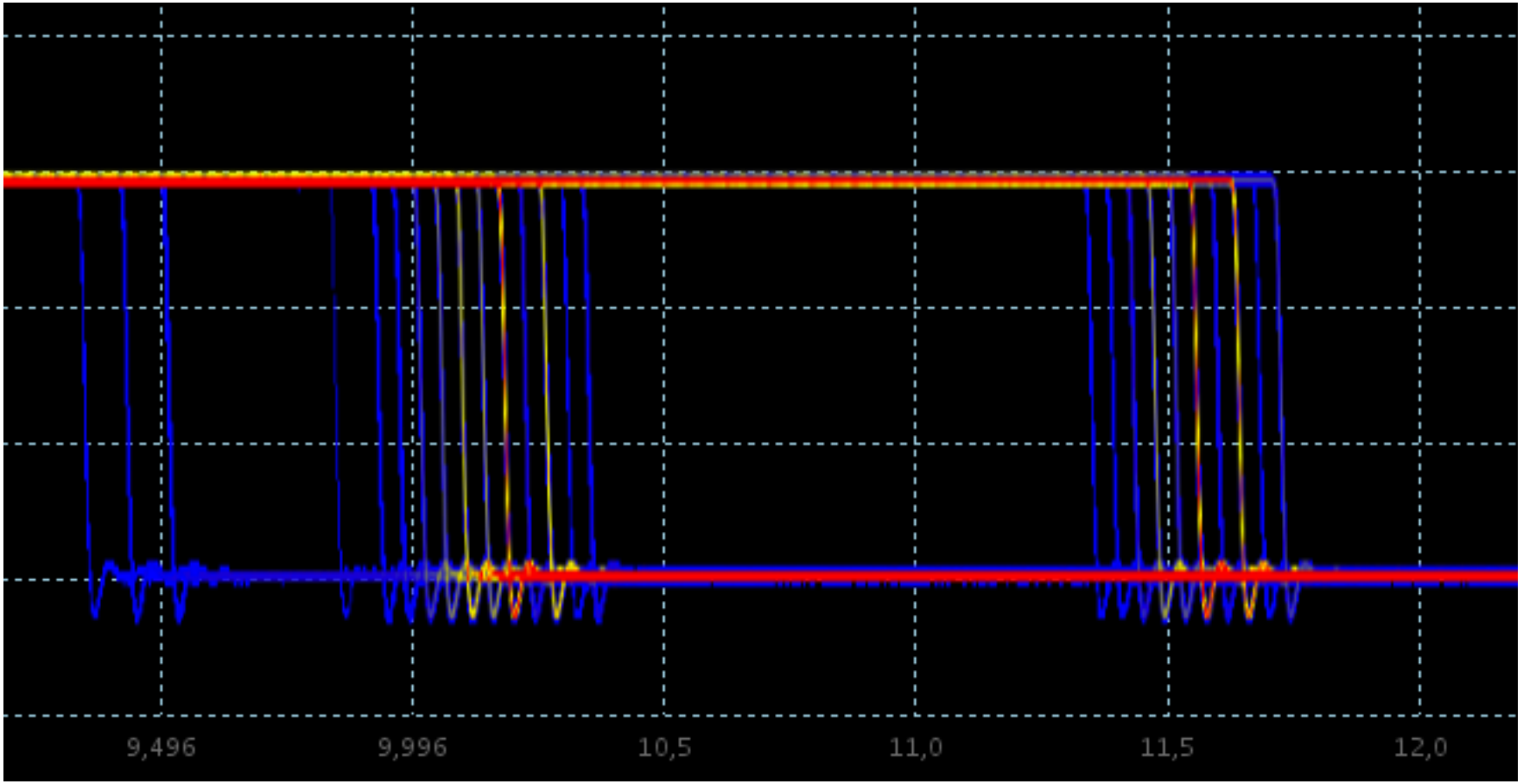
Perspectives

- Implantation d'un AES permettant plus de combinaisons
  - Boucles déroulées
  - Calcul de la SBOX, pas d'accès mémoire
- Attaque en fautes
  - Intérêt de la protection par allocation de registre ↗
  - Faute lors de la génération du code polymorphique pour obtenir une instance unique
- Rétroconception
  - Déterminer le code assembleur

Analyse sécuritaire

- Attaque par corrélation de Pearson (CPA)
- Tri par occurrence temporelle et CPA
- Tri par Magnitude Squared Coherence et CPA

Protection	CPA	# courbes / SR	CPA (sélection)	# courbes	CPA (MSC)	# courbes
Sans	**'AA' / 0.8**	50 / ≈60%				
RA	**'AA' / 0,11**	5000	**'AA' / 0,83**	45 / **500**	**'AA' / -0,38**	262 / **500**
IS	**'AA' / 0,3**	5000	**'AA' / 0,8**	42 / **200**		
S	**'AA' / 0,7**	5000	**'AA' / 0,8**	52 / **100**	**'AA' / 0,7**	100 / **100**
Noise	**'87' / 0.08**	5000	NA	5000	**'AA' / 0,17**	1114 / **5000**
All	**'58' / 0.06**	7000	NA	5000	**'F9' / 0.13**	1251 / **5000**


9,496
9,996
10,5
11,0
11,5
12,0