김현지

https://youtu.be/QwPoLBUVhO4





# Contents

01. AES

02. 동작

03. 구현

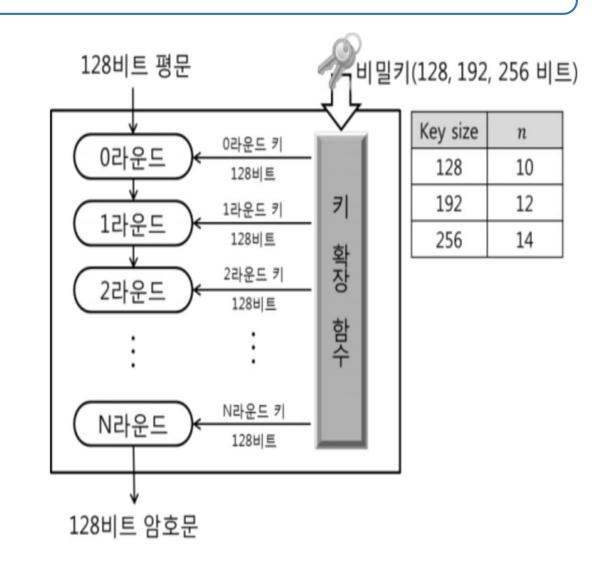


01. 배경



# AES 구조

- ❖ AES 128 / 192 / 256 (key의 길이)
  - > 각각 10 / 12 / 14 round
- ❖ 블록 암호 & SPN 구조
  - ➤ 블록 크기 = 128bits
  - ▶ 병렬 연산 가능 (속도 빠름)
  - ▶ 별도의 복호화 모듈 필요
- ❖ 128 bit Key로 한번에 암호화
- ❖ DES에 비해 key의 길이, 블록 크기 증가 / round 감소
- ❖ State
  - ➤ 16 byte (=128 bits) 의 4x4 행렬
  - Block이 state로 표현



## AES 수학적 배경

#### ❖ GF(28) 상에서의 확장체 연산

- AES에서 유한체는 256개의 원소를 포함 → 체의 원소 각각이 하나의 Byte로 표현 가능
  - $\triangleright$  각 원소는 다항식 형태로 표현 : A  $\in$  GF(2<sup>8</sup>), A(x) =  $x^7 + x^6 + x^3 + x + 1 = 11001011$
- 계산 결과가 GF(28)상의 원소가 되도록 연산
  - ▶ 이 때, mod 256이 아닌 mod P(x) 상에서의 연산 : P(x) = x<sup>8</sup> + x<sup>4</sup> + x<sup>3</sup> + x + 1
- GF(28)상의 역원 : Subbyte 과정에서 쓰이는 S-box는 각 원소에 대한 역원을 미리 계산한 Lookup Table

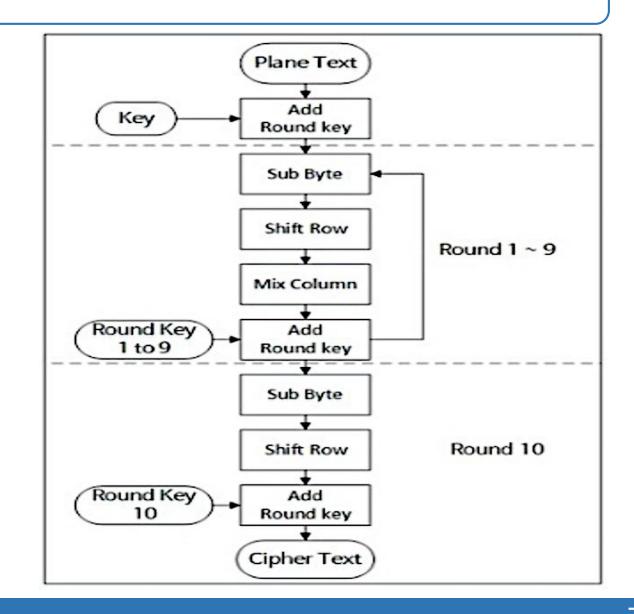


02. 동작



#### ❖동작원리

- 각 round에는 4가지 연산
  - Sub Byte / Shift Row / Mix Column / Add Round Key
  - 치환 연산 3개, 단순 shift1개
- 0 round : Add Round-key (preprocessing)
- 10 round : Mix column 수행 x
- Round-key는 키 확장 함수에 의해 생성
  - Round마다 다른 key 사용
  - round-key: 128bit



## ❖Plain text(128 bit) → State

Ι	Α	Ν	S	U	N	G	כ	Ν		V	Ш	R	S		Т
48	41	4E	53	55	4E	47	55	4E	49	56	45	52	53	49	54

48	55	4E	52
41	4E	49	53
4E	47	56	49
53	55	45	54

Plain Text



#### **❖0-round : Add Round-Key**

48	55	4E	52
41	4E	49	53
4E	47	56	49
53	55	45	54

Plain Text

 $\oplus$ 

11	55	4F	F0
22	44	A2	B1
33	11	9D	33
44	22	3C	52

Cipher Key

59	00	01	A2
63	OΑ	EB	E2
7D	56	СВ	7A
17	77	79	06

Add Round-key Output



## ❖Sub Byte

59	00	01	A2
63	OA	EB	E2
7D	56	СВ	7A
17	77	79	06

										y							
		0	1	2	3	4	5	6	7	8	9	A	В	С	D	Е	F
	0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
	1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
	2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
	3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
	4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	В3	29	E3	2F	84
	5	53	Dl	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
	6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
ı,	7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
*	8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
	9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
	A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
	В	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
	С	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
	D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	Cl	1D	9E
	E	El	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
	F	8C	Al	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

СВ	63	7C	3A
FB	67	<b>E</b> 9	98
FF	B1	1F	DA
F0	F5	B6	6F

Sub byte Input

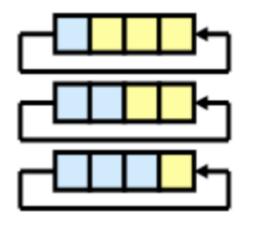
S-box

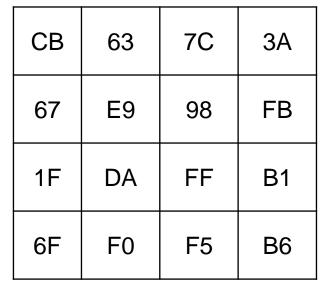
Sub Byte
Output



#### **♦** Shift-Row

СВ	63	7C	ЗА
FB	67	E9	98
FF	B1	1F	DA
F0	F5	В6	6F





Shift-row Input Shift-Row Output



#### **♦ Mix-Column**

02	03	01	01
01	02	03	01
01	01	02	03
03	01	01	02

Mix-Column Matrix

СВ	63	7C	3A
67	<b>E</b> 9	98	FB
1F	DA	FF	B1
6F	F0	F5	B6

\*

State

54	CC	41	65
4B	2F	B8	A9
23	2E	05	79
E0	6D	12	73

Mix-Column
Output





- 1 \* input
  - > input
- 2 \* input
  - ➤ Overflow 발생 시, 최상위 비트는 떼어버림 (9 → 8-bits)
    - → 해당 결과 값 ⊕ 0x1B
- 3 \* input
  - > 1 \* input + 2 \* input
- \*\* 0x1B를 XOR하는 이유?
  - : GF(28)상에서의 기약다항식 P(x) = x8 + x4 + x3 + x + 1
    - ▶ overflow 발생 (8bit초과) 시, x<sup>8</sup> 이상의 다항식 항은 확장체 나눗셈 연산 통해 제거
    - ➤ 곱해지는 값이 최대 03 (= x + 1)이므로 x<sup>8</sup>을 초과하지 않음

    - $\triangleright$  00011011 = 0x1B

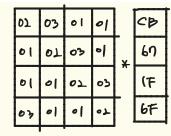
65

**A9** 

79

73

#### ❖Mix-Column 다항식 계산



7) 
$$CB = 11001011 = \pi^{11} + \chi^{6} + \chi^{3} + \chi^{4}$$
 $0L = 00000010 = \pi$ 
 $0L \times CB = \chi(\chi^{11} + \chi^{6} + \chi^{3} + \chi^{4} + \chi^{4}$ 

\* 0x1B XOR 이용하여 계산한 값과 동일

54	CC	41	65
4B	2F	B8	A9
23	2E	05	79
E0	6D	12	73

Mix-Column
Output

## **❖Add Round-key**

54	CC	41	65
4B	2F	B8	A9
23	2E	05	79
E0	6D	12	73

 $\oplus$ 

D8	8D	C2	32
E1	A5	07	B6
33	22	BF	8C
C8	EA	D6	84

=

8C	41	83	57
AA	8A	BF	1F
10	0C	ВА	F5
28	87	C4	F7

Add Round-key

Input

Round-key

Round 종료



## Key Scheduling

11	55	4F	F0
22	44	A2	B1
33	11	9D	33
44	22	3C	52
1			1

Cipher Key

B1	
33	
52	
F0	

Rotation

			y														
		0	1	2	3	4	5	6	7	8	9	A	В	С	D	Е	F
	0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
	1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
	2	B7	FD	93	26	36	3F	<b>F</b> 7	CC	34	A5	E5	F1	71	D8	31	15
	3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
	4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	В3	29	E3	2F	84
	5	53	Dl	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
	6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
	7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
x	8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
	9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
	A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
	В	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
	С	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
	D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
	Е	El	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
	F	8C	Al	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

	C8
	C3
=	00
	8C

S-box Sub Byte Output

## Key Scheduling

\* i round의 Round key 구하는 경우

1 round



		_													_			
	C8		01	02	04	08	10	20	40	80	1b	36		C9		11		D8
	C3		00	00	00	00	00	00	00	00	00	00		C3	(	22	_	E1
	00	$\oplus$	00	00	00	00	00	00	00	00	00	00	=	00	$\oplus$	33	=	33
	8C		00	00	00	00	00	00	00	00	00	00		8C		44		C8
5	Sub Byte	; ;					Do	00						Rcon	' 이전	(i-1) roui	nd의	i round⊆

Sub Byte

Output

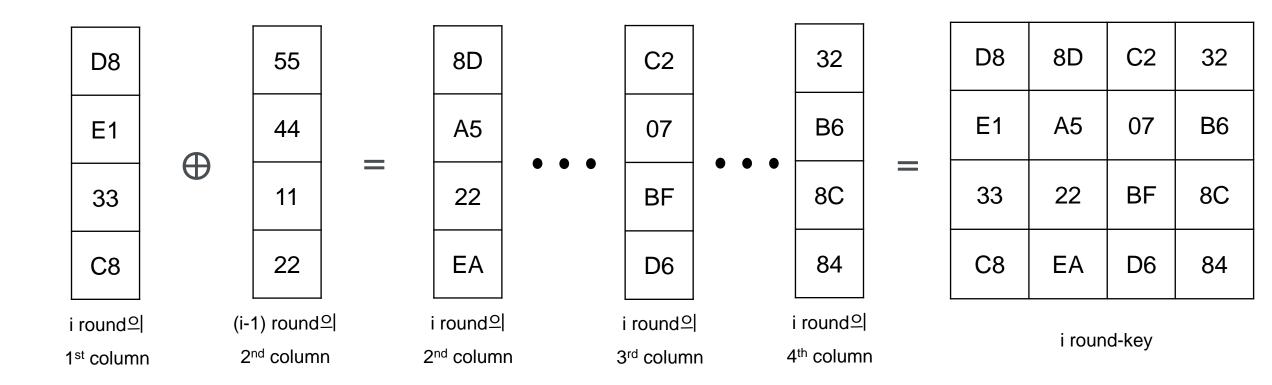
Rcon

Rcon 이전(i-1) round의 output 1<sup>st</sup> column i round의

1<sup>st</sup> column



#### Key Scheduling



03. 구현



u8 w[176];

w[16*i+j+0]	w[]	w[]	w[]
w[16*i +j+4]	w[]	w[]	w[]
w[16*i+j+8]	w[]	w[]	w[]
w[16*i+j+12]	w[]	w[]	w[]

```
for(int j = 1; j < 4; j++){

w[16*i + (0+j)] = w[16*(i-1) + 0+j] ^ w[16*i + (j-1) + 0]; //

w[16*i + (4+j)] = w[16*(i-1) + 4+j] ^ w[16*i + (j-1) + 4]; //

w[16*i + (8+j)] = w[16*(i-1) + 8+j] ^ w[16*i + (j-1) + 8];

w[16*i + (12+j)] = w[16*(i-1) + 12+j] ^ w[16*i + (j-1) + 12];
}</pre>
```

```
for(int i = 1 ; i < 11 ; i++){
    // 이전 라운드의 마지막 열을 연산에 사용하기 위해 temp[]에 저장
    temp[0] = w[16 * (i-1) + 4*0 + 3]; // 16 * (i-1) --> 이전 round key의 맨 처음 요소 :w[0],w[16]...
    temp[1] = w[16 * (i-1) + 4*1 + 3]; // 4*1, 4*2...-> 이전 round key의 맨 처음 열, +3 --> 마지막 열
    temp[2] = w[16 * (i-1) + 4*2 + 3]; |
    temp[3] = w[16 * (i-1) + 4*3 + 3];
```

w[16*(i-1)]	w[]	w[]	w[16*(i-1) + 4*0 +3]
w[16*(i-1) + 4*1]	w[]	w[]	w[16*(i-1) + 4*1 +3]
w[16*(i-1) + 4*2]	w[]	w[]	w[16*(i-1) + 4*2 +3]
w[16*(i-1) + 4*3]	w[]	w[]	w[16*(i-1) + 4*3 +3]

w[16]	w[17]	w[18]	w[19]
w[20]	w[21]	w[22]	w[23]
w[24]	w[25]	w[26]	w[27]
w[28]	w[29]	w[30]	w[31]

\* 2 round : w[32]~ w[47] → 1 round의 마지막 열 사용



Input = 32 43 f6 a8 88 5a 30 8d 31 31 98 a2 e0 37 07 34

Cipher Key = 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c

The Round Key values are taken from the Key Expansion example in Appendix A.

ound Ke	y values are ta	iken from the K	Key Expansion	example in A	ppendix A.
Round Number	Start of Round	After SubBytes	After ShiftRows	After MixColumns	Round Key Value
	32 88 31 e0				2b 28 ab 09
input	43 5a 31 37				⊕ 7e ae f7 cf =
•	f6 30 98 07				15 d2 15 4f
	a8 8d a2 34				16 a6 88 3c
	19 a0 9a e9	d4 e0 b8 1e	d4 e0 b8 1e	04 e0 48 28	a0 88 23 2a
	3d f4 c6 f8	27 bf b4 41	bf b4 41 27	66 cb f8 06	fa 54 a3 6c
1	e3 e2 8d 48	11 98 5d 52	5d 52 11 98	81 19 d3 26	⊕ fe 2c 39 76 =
	be 2b 2a 08	ae f1 e5 30	30 ae f1 e5	e5 9a 7a 4c	17 b1 39 05
2	a4 68 6b 02	49 45 7f 77	49 45 7f 77	58 1b db 1b	f2 7a 59 73
	9c 9f 5b 6a	de db 39 02	db 39 02 de	4d 4b e7 6b	c2 96 35 59
	7f 35 ea 50	d2 96 87 53	87 53 d2 96	ca 5a ca b0	⊕ 95 b9 80 f6 =
	f2 2b 43 49	89 f1 la 3b	3b 89 f1 1a	f1 ac a8 e5	f2 43 7a 7f
	12 23 13	03 11 14 32	35 05 11 14	11 40 40 65	12 43 74 72
	aa 61 82 68	ac ef 13 45	ac ef 13 45	75 20 53 bb	3d 47 1e 6d
3	8f dd d2 32	73 c1 b5 23	c1 b5 23 73	ec 0b c0 25	⊕ 80 16 23 7a =
	5f e3 4a 46	cf 11 d6 5a	d6 5a cf 11	09 63 cf d0	47 fe 7e 88
	03 ef d2 9a	7b df b5 b8	b8 7b df b5	93 33 7c dc	7d 3e 44 3b
4	48 67 4d d6	52 85 e3 f6	52 85 e3 f6	0f 60 6f 5e	ef a8 b6 db
	6c 1d e3 5f	50 a4 11 cf	a4 11 cf 50	d6 31 c0 b3	44 52 71 0b
	4e 9d b1 58	2f 5e c8 6a	c8 6a 2f 5e	da 38 10 13	⊕ a5 5b 25 ad =
	ee 0d 38 e7	28 d7 07 94	94 28 d7 07	a9 bf 6b 01	41 7f 3b 00
	e0 c8 d9 85	e1 e8 35 97	e1 e8 35 97	25 bd b6 4c	d4 7c ca 11
	92 63 b1 b8	4f fb c8 6c	fb c8 6c 4f	d1 11 3a 4c	d1 83 f2 f9
5	7f 63 35 be	d2 fb 96 ae	96 ae d2 fb	a9 d1 33 c0	⊕   d1   63   12   19   =
	e8 c0 50 01	9b ba 53 7c		ad 68 8e b0	f8 87 bc bc
	eo CU 50 01	3D Da 33 /C	7c 9b ba 53	ad 08 8e D0	18 8 / DC DC

	f1	c1	7с	5d		a1	78	10	4c		a1	78	10	4c		4b	2c	33	37		6d	11	đb	ca	
6	00	92	с8	<b>b</b> 5		63	4f	<b>e</b> 8	d5 1 03		4f	е8	8 d5 63		86	4a 9d		d2		88	0Ъ	f9	00	=	
	6£	4c	8b	d5		a8 :	29	3d			3d	03	<b>a</b> 8	29		8d	89	f4	18	<b>⊕</b>	<b>a</b> 3	3е	86	93	
	55	ef	32	0с		fc	df	23	fe		fe	fc	df	23		6d	80	е8	d8		7a	fd	41	fd	
										•										l					
	26	3d	<b>e</b> 8	fd		f7	27	9b	13 b5 10 3d		£7	£7 27		b5 ab 31 a9		14 46 15 16	46	27	34		4e	5f	84	4e	
7	0e	41	64	d2		ab	83	43			83	43					46 2a	ω	54	5f	<b>a</b> 6	<b>a</b> 6	=		
	2e	b7	72	8b		31	<b>a</b> 9	40				3d 3	31		l L	b5	15		<b>⊕</b>	£7	с9	4f	dc	_	
	17	7d	<b>a</b> 9	25		f0	0 ff d	d3					ff			bf	ec			0e	f3	b2	4f		
										•										l					
	5 <b>a</b>	19	<b>a</b> 3	7a		be	d4	0a	da		be	d4	0a	da		00	b1	54	fa		ea	b5	31	7£	
8	41	49	e0	8c		83	3b	e1	64		3b	e1	64	83		51	с8	76	1b	<b>⊕</b>	d2	8d	2b	8d	=
	42	dc	19	04			86	d4	f2		d4	f2	2c	86		2f	89	6d 99	99	θ	73	ba	f5	29	
	b1	1f	65	0с			c0	4d	fe	İİ	fe	с8	c0	4d	d d1	d1	ff	cd	ea		21	d2	60	2f	
		·								•															
	ea	04	65	85		87	f2	4d	97		87	f2	4d 90	97 ec		47	40	<b>a</b> 3	4c		ac	19	28	57	7
				96		ec	6e 4c	40	90		6e	4c				37	d4	70	9f		77	fa	d1	5с	
a	83	45	5d			-			-		اوو									Д	l′′				=
9		45 33	5d 98			4a	с3	46	<b>e</b> 7		46	<b>e</b> 7	4a	с3		94	e4	3 <b>a</b>	42	<b>⊕</b>	66	dc	29	00	=
9	5с		98			4a	c3 d8	46				e7 8c		c3 95		94 ed	e4 a5	3a a6		<b>⊕</b>			29 41		H
9	5с	33	98	ъ0		4a		46	<b>e</b> 7		46									<b>⊕</b>	66	dc		00	=
9	5c	33	98	ъ0		4a	<b>d8</b>	46	e7 a6		46		<b>d8</b>	95						<b>⊕</b>	66	dc		00 6e	=
	5c f0	33 2d	98 ad	b0 с5		4a 8c	<b>d8</b>	46 95	e7 a6		46 a6	8c	<b>d8</b>	95							66 £3	dc 21	41	00 6e	=
9	5c f0 eb	33 2d 59	98 ad 8b	b0 c5		4a 8c e9	d8	46 95 3d	e7 a6		46 a6	8c cb	d8 3d	95 af						<ul><li>+</li><li>+</li><li>+</li></ul>	66 f3	dc 21 c9	41 e1	00 6e b6	
	5c f0 eb 40 f2	33 2d 59 2e	98 ad 8b a1	b0 c5 1b		4a 8c e9	cb 31 07	46 95 3d 32	e7 a6 af 2e 2c		46 a6 e9	8c cb 32	3d 2e 89	95 af 09							66 f3 d0	dc 21 c9 ee	41 e1 3f	00 6e b6 63	
	5c f0 eb 40 f2	33 2d 59 2e 38	98 ad 8b a1 13	b0 c5 1b c3 42		4a 8c e9 09	cb 31 07	46 95 3d 32 7d	e7 a6 af 2e 2c		46 a6 e9 31 7d	8c cb 32 2c	3d 2e 89	95 af 09							66 f3 d0 14 f9	dc 21 c9 ee 25	41 e1 3f 0c	00 6e b6 63 0c	
	5c f0 eb 40 f2 1e	33 2d 59 2e 38 84	98 ad 8b a1 13	b0 c5 1b c3 42		4a 8c e9 09	cb 31 07	46 95 3d 32 7d	e7 a6 af 2e 2c		46 a6 e9 31 7d	8c cb 32 2c	3d 2e 89	95 af 09							66 f3 d0 14 f9	dc 21 c9 ee 25	41 e1 3f 0c	00 6e b6 63 0c	
10	5c f0 eb 40 f2 1e	33 2d 59 2e 38 84	98 ad 8b a1 13 e7	b0 c5 1b c3 42 d2		4a 8c e9 09 89 72	cb 31 07 5f	3d 32 7d 94	e7 a6 2e 2c b5		46 a6 31 7d b5	2c 72	3d 2e 89	95 af 09							66 f3 d0 14 f9	dc 21 c9 ee 25	41 e1 3f 0c	00 6e b6 63 0c	
	5c f0 40 f2 1e	33 2d 59 2e 38 84	98 ad 8b a1 13 e7	b0 c5 lb c3 42 d2		4a 8c e9 09 89 72	cb 31 07 5f	3d 32 7d 94	e7 a6 2e 2c b5	ec	46 a6 e9 31 7d	2c 72	3d 2e 89	95 af 09							66 f3 d0 14 f9	dc 21 c9 ee 25	41 e1 3f 0c	00 6e b6 63 0c	

# Q&A

