AES CTR Algorithm

Marco Rando S4249928
Riccardo Bianchini S4231932

Università degli Studi di Genova

February 15, 2020

Summary

What is AES?

AES: Definition

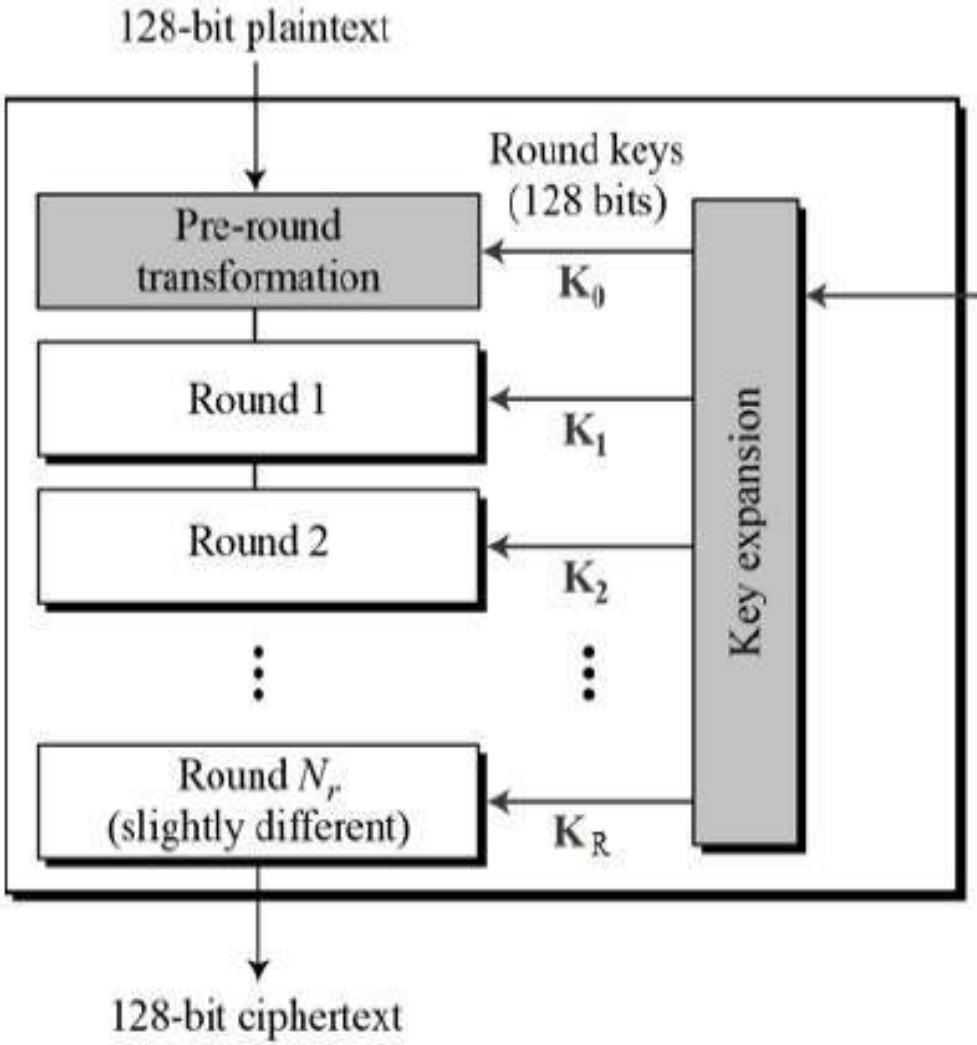
AES (Advanced Encryption Standard) is a specification for the encryption of data.

- Simmetric key
- Subset of the Rijndael block cipher

In nowdays, three version of AES exist: 128, 192 and 256 (this number indicates the size of the block).

What is AES?

AES in general

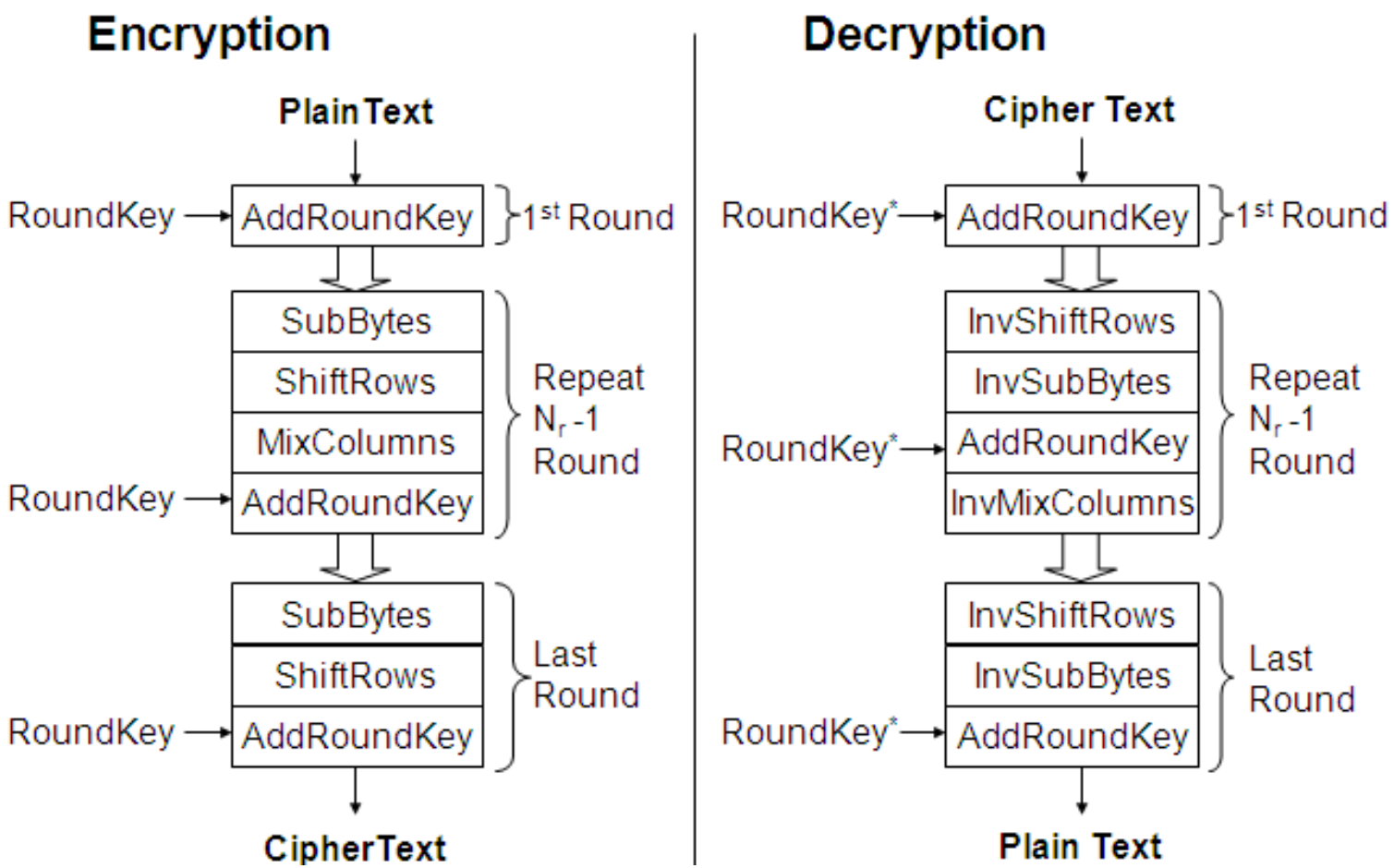


R	Key size
10	128
12	192
14	256

Relationship between number of rounds (R) and cipher key size

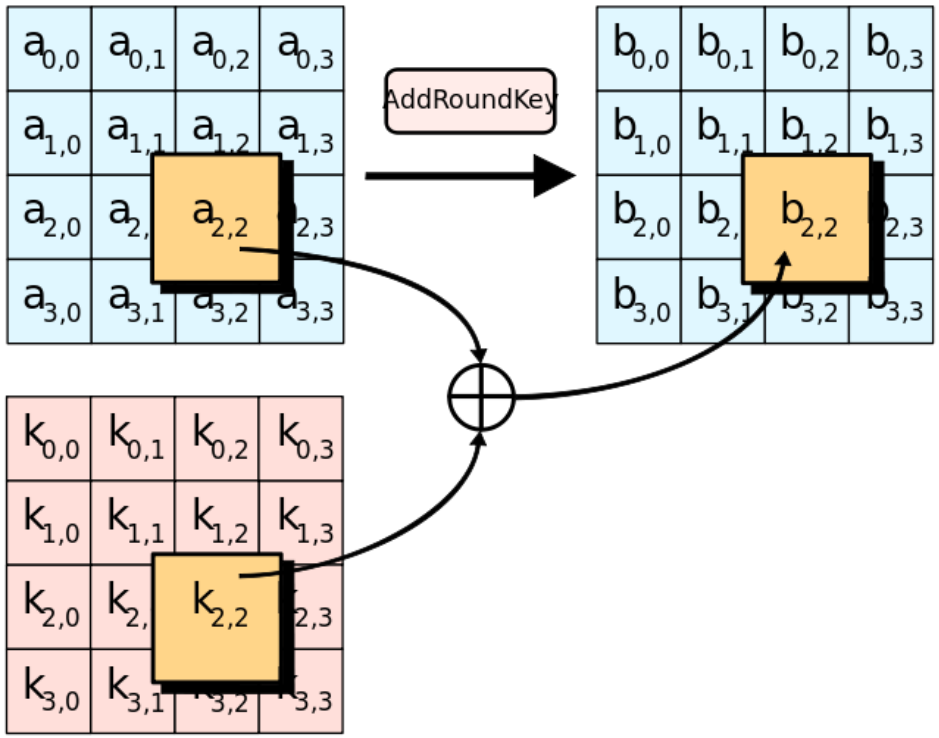
What is AES?

AES: operations



Round Key Phase

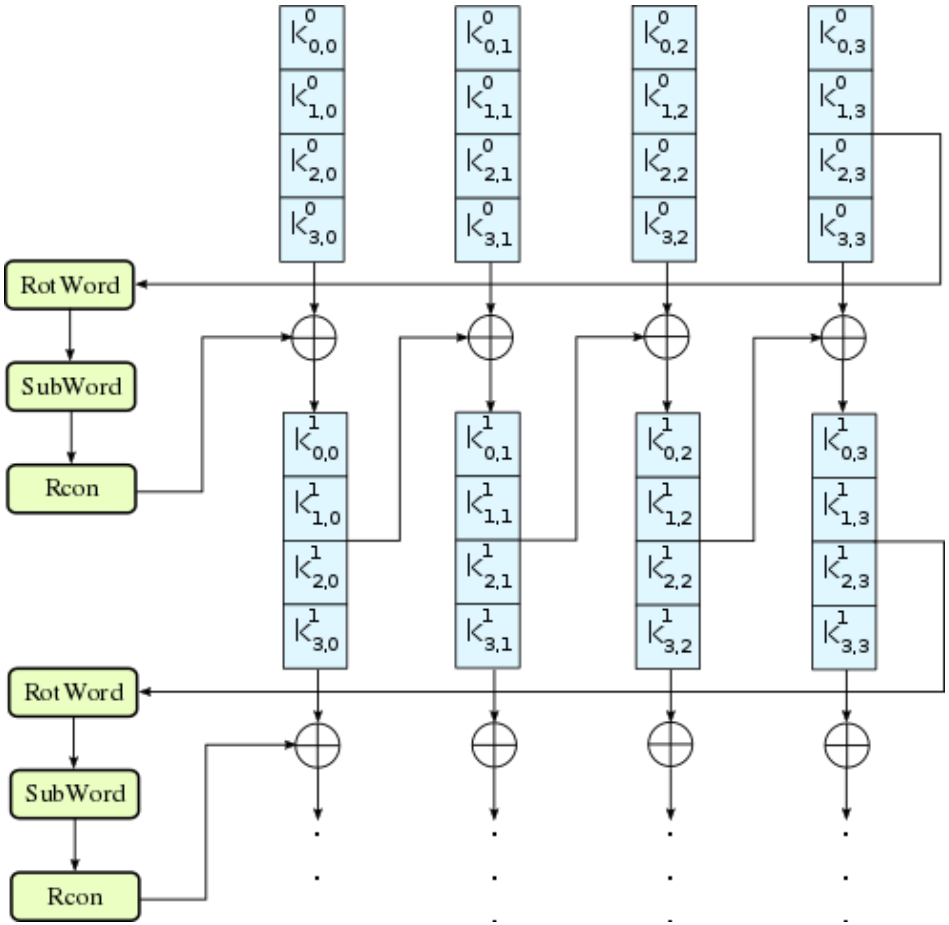
AES: Round Key



- Different key (called sub-keys) for different round.
- Sub-keys generated using a key schedule algorithm.

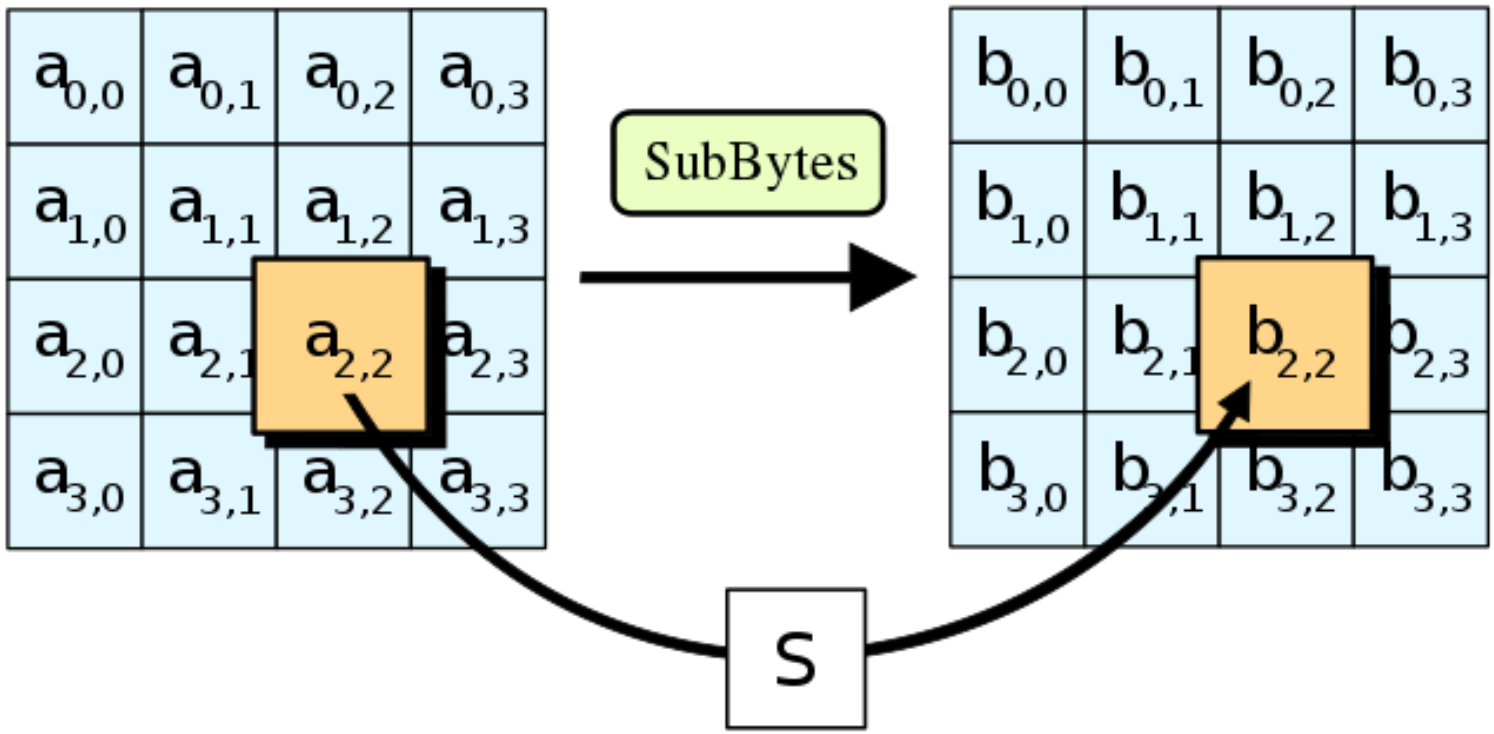
Rijndael's Key Schedule

AES: Key Schedule Algorithm



Sub Byte Operation

AES: Sub Byte



S-Box and RS-Box

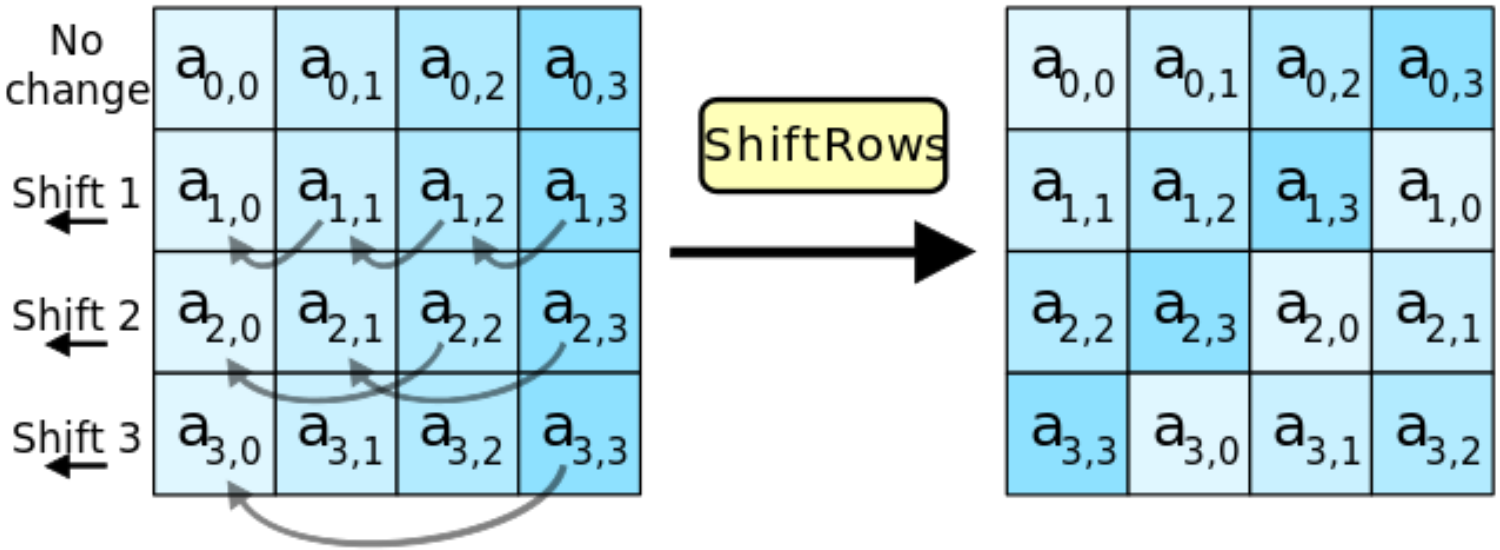
AES: S-Box and RS-Box

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
10	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
20	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
30	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
40	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
50	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
60	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
70	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
80	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
90	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
a0	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
b0	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
c0	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
d0	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
e0	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
f0	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Shift Row Operation

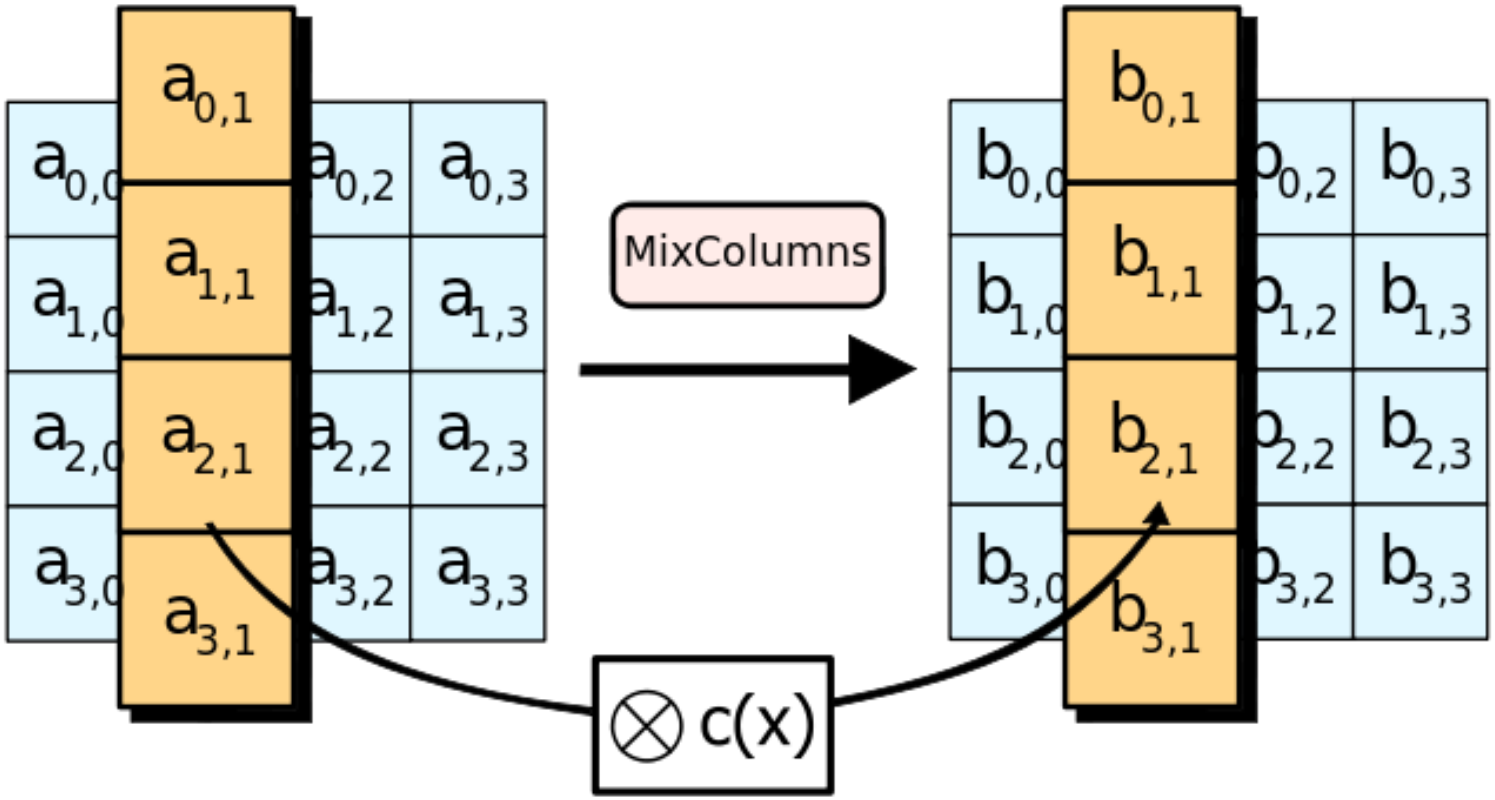
AES: Shift Row



- Shift (to left) row i by i positions.

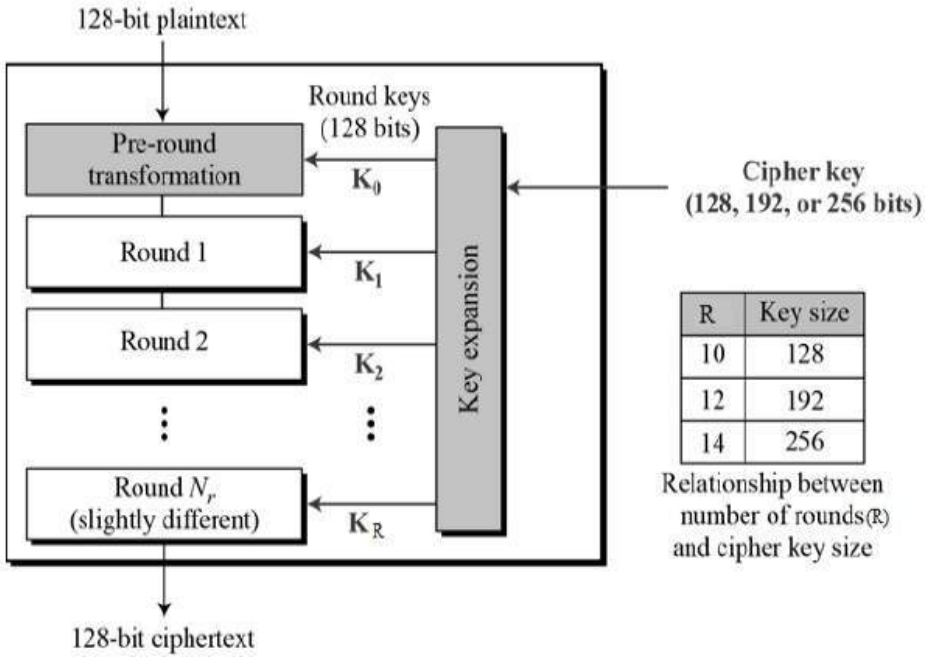
Mix Column Operation

AES: Mix Column



AES: Different Text Dimensions

AES: Different Dimensions



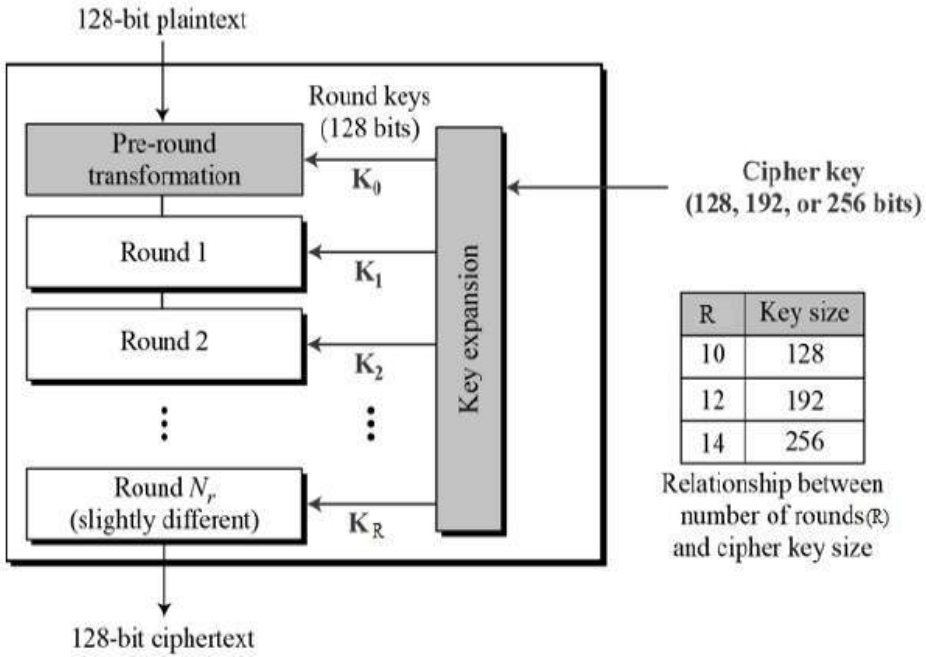
R	Key size
10	128
12	192
14	256

Relationship between number of rounds(R) and cipher key size

- And if we want to encrypt and decrypt plain texts larger/smaller than 16 byte?

AES: Different Text Dimensions

AES: Different Dimensions



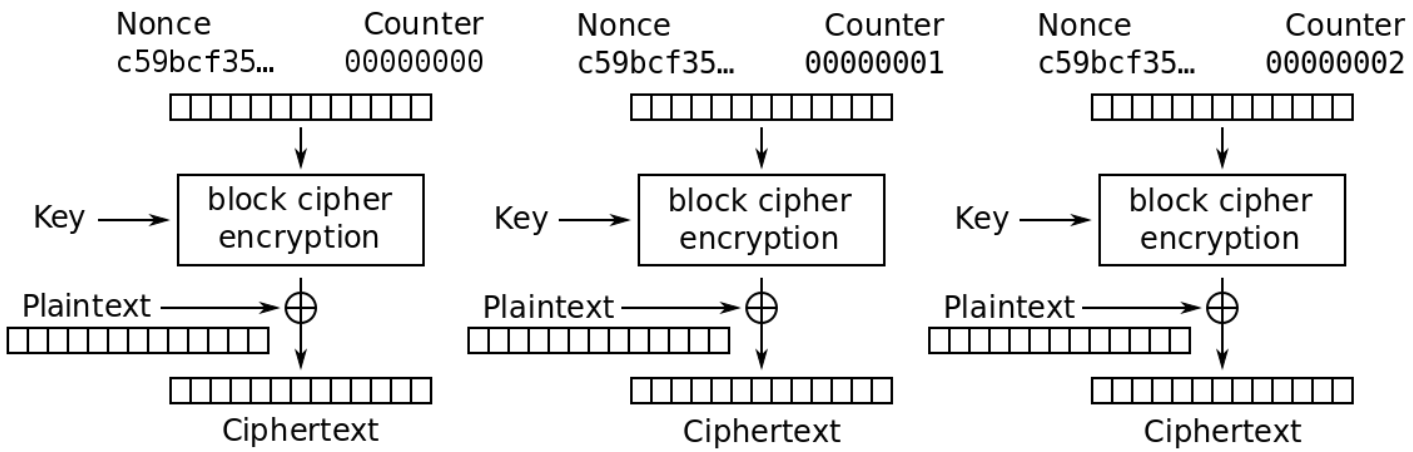
R	Key size
10	128
12	192
14	256

Relationship between number of rounds(R) and cipher key size

- And if we want to encrypt and decrypt plain texts larger/smaller than 16 byte?
  - If smaller: we can add padding
  - If bigger: we need a modality to "concat" encrypted blocks

CTR Mode

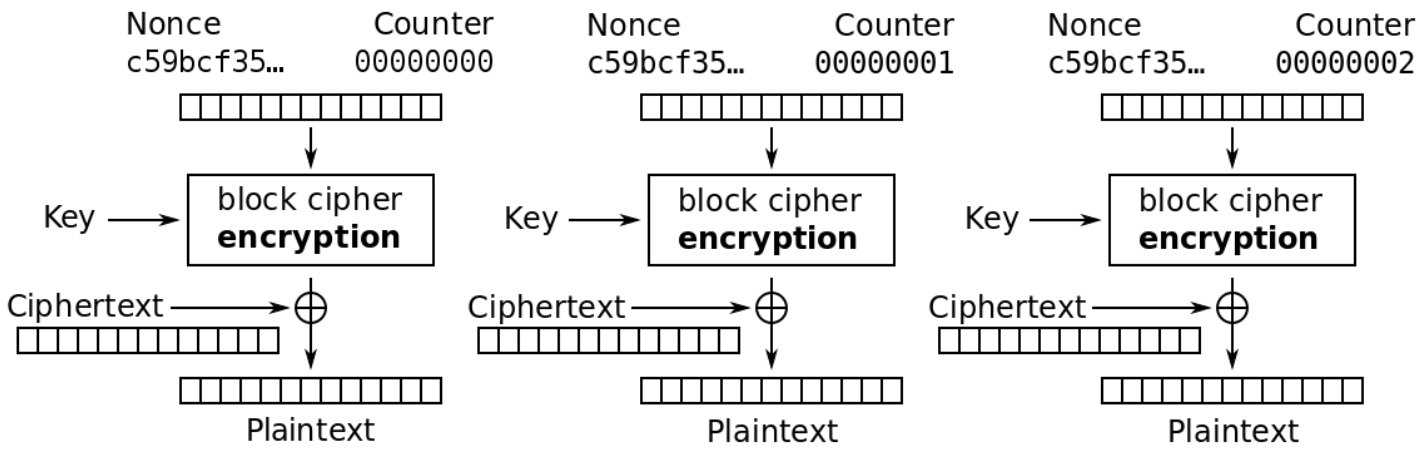
CTR: Encryption



Counter (CTR) mode encryption

CTR Mode

CTR: Decryption



Counter (CTR) mode decryption

AES CTR: Example

AES CTR executions

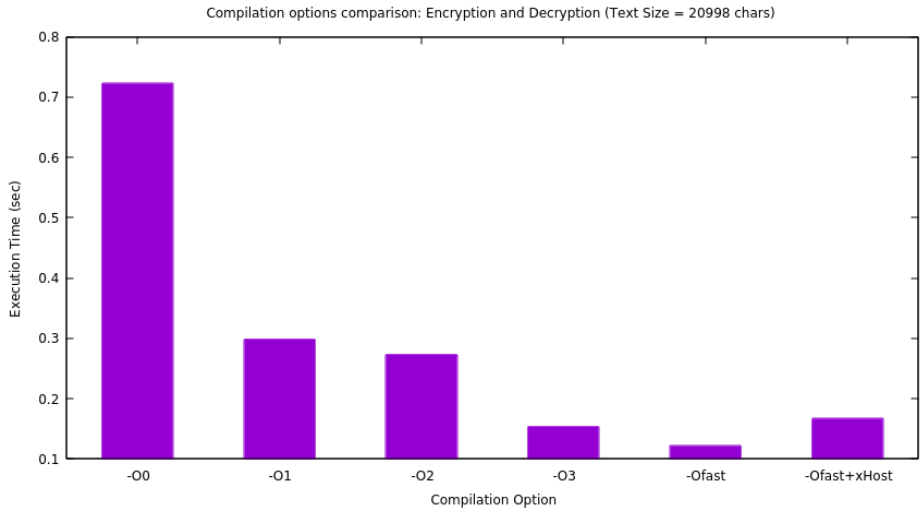
```
[--] Plain Text: V3ryS3cr3tT3xt

[--] CTR encripted: 67f9a60fc583e7c7ec03d6a826c7a986
[--] CTR Decripted: V3ryS3cr3tT3xt
```

```
[--] Plain Text: 50r3h4,himi7sud3desuよ !

[--] CTR encripted: 04faa645fe84a8ddb61aebac2dc6c7b5476dd955458a71303e307352c92953f8
[--] CTR Decripted: 50r3h4,himi7sud3desuよ !
```

```
[--] Plain Text: 秘密

[--] CTR encripted: d66d4c9339368eb5df77829b5eb3a386
[--] CTR Decripted: 秘密
```

Sequential Version

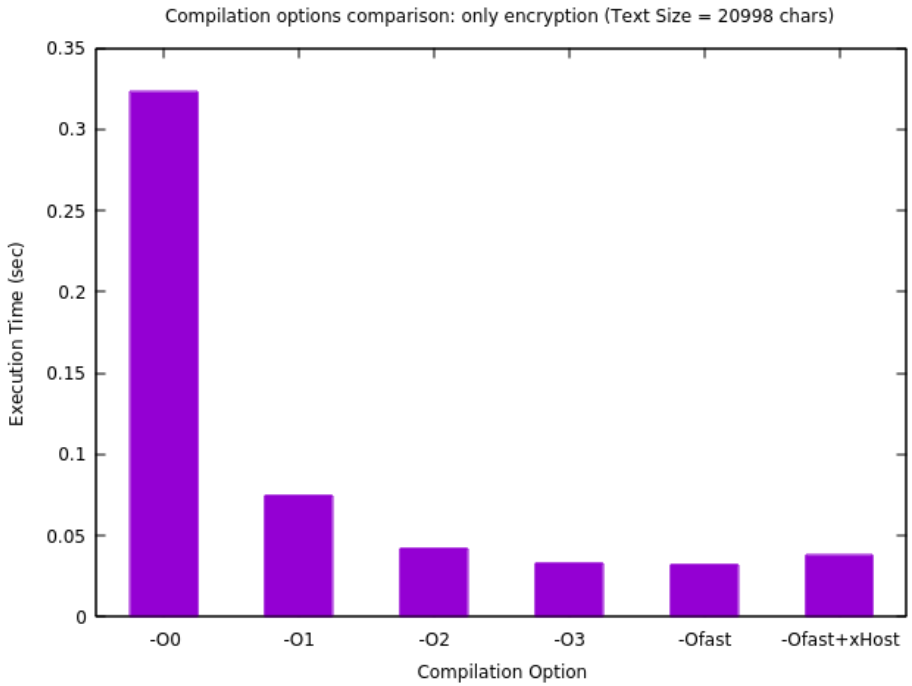
Compilation Options



- Execution on 1312 blocks.
- xHost option worsen performances: expensive operations non-vectorizable.

Sequential Version

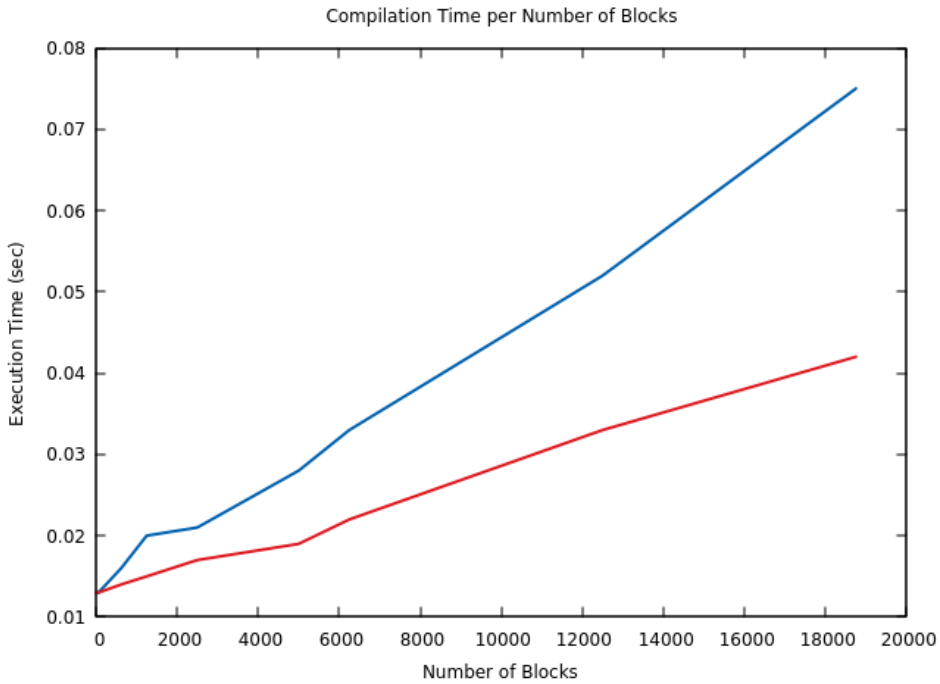
Compilation Options: why don't only encryption?



- **Times are too low**.
- It is $\sim \frac{1}{2}$ of encryption and decryption.

Sequential Version

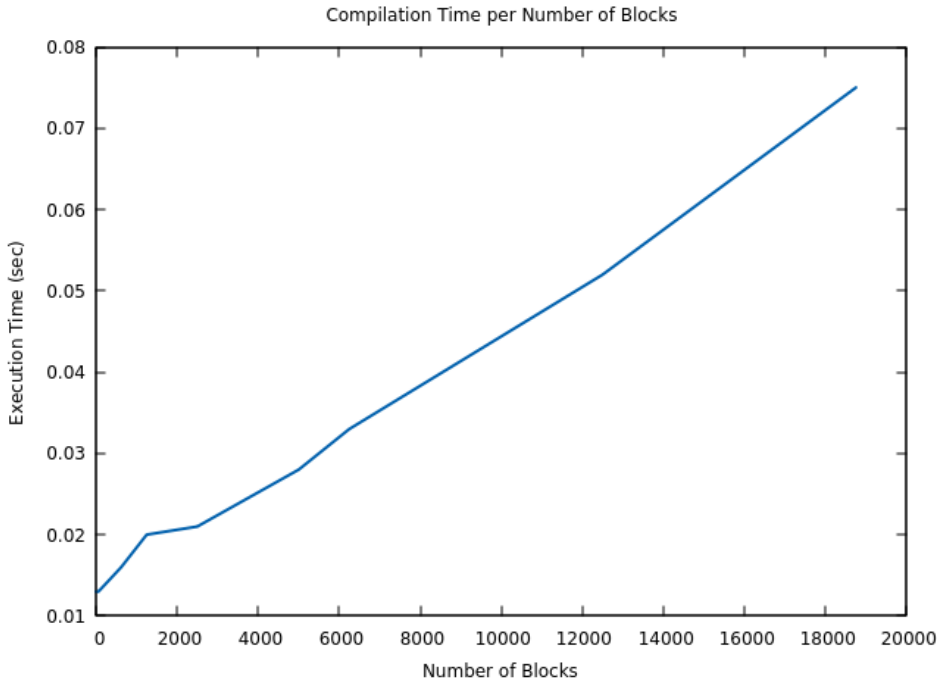
Compilation Options: why don't only encryption?



- Times are too low.
- **It is $\sim \frac{1}{2}$ of encryption and decryption.**

Sequential Version

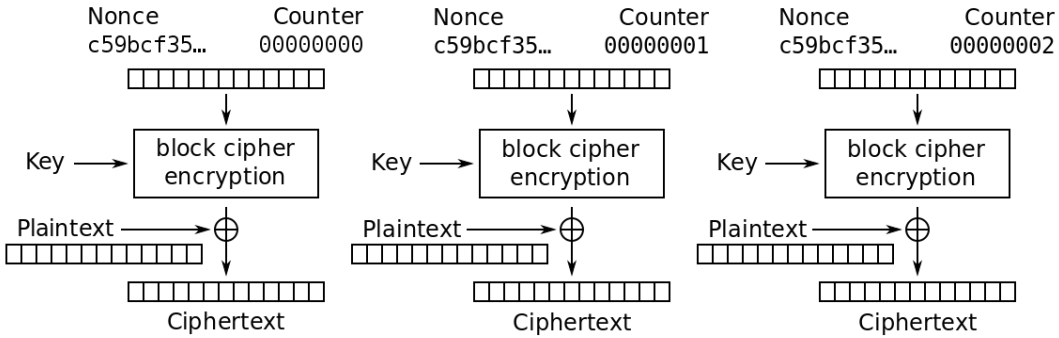
Sequential: Execution time per Number of blocks



- Obviously, it grows linearly with respect number of blocks.

OpenMP: What should we parallelize?

AES CTR

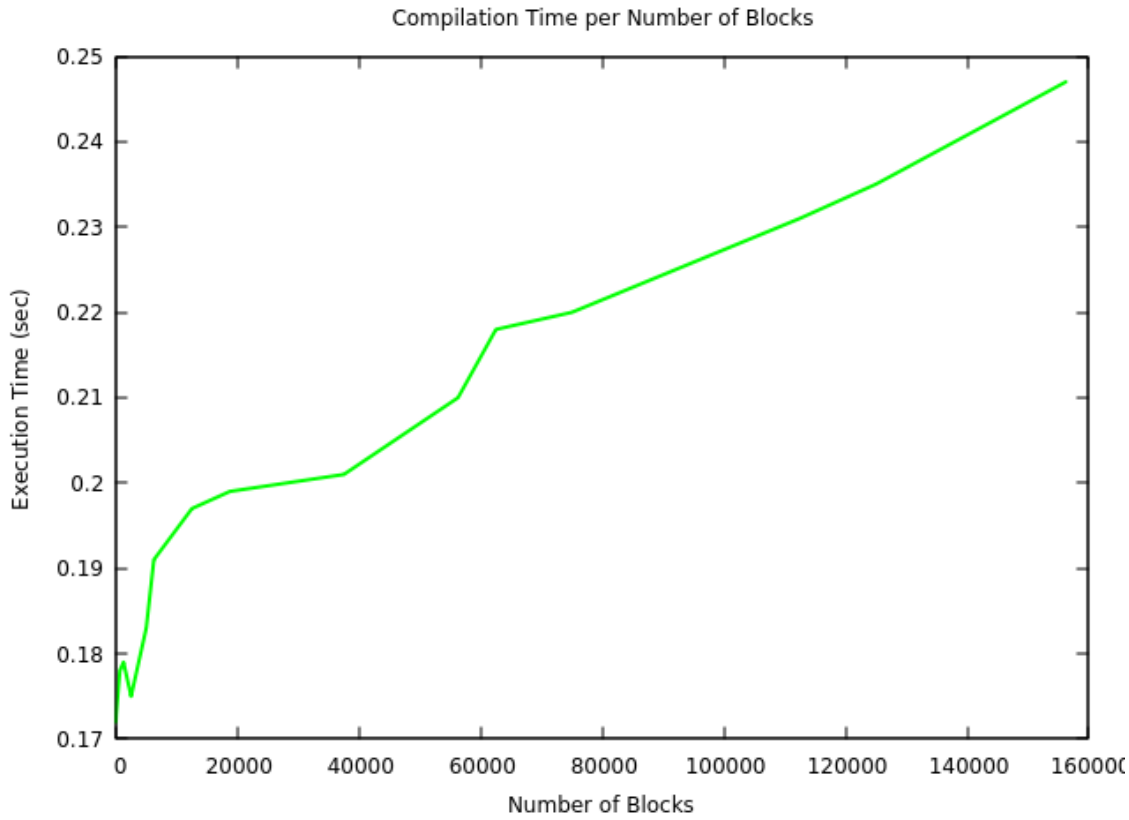


Counter (CTR) mode encryption

- No parallel region inside AES.
- Parallelize the CTR Mode.
- Sub-keys created by a single thread.

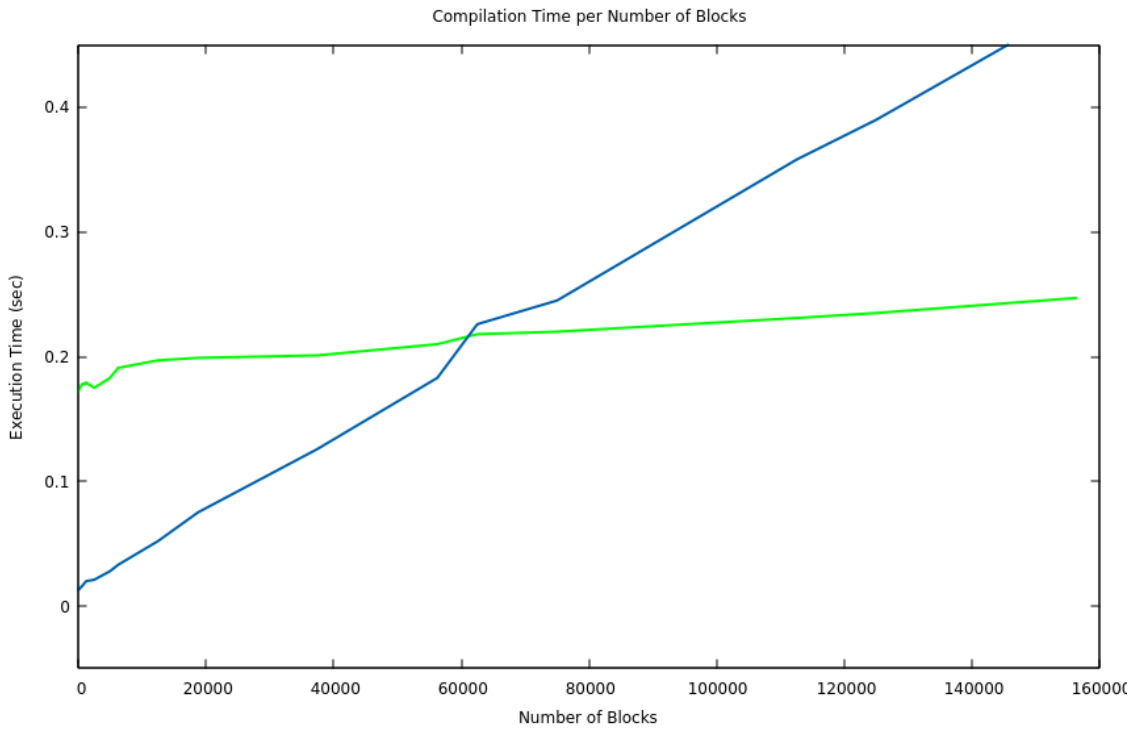
OpenMP: Execution time with default schedule

OpenMP: Execution Time



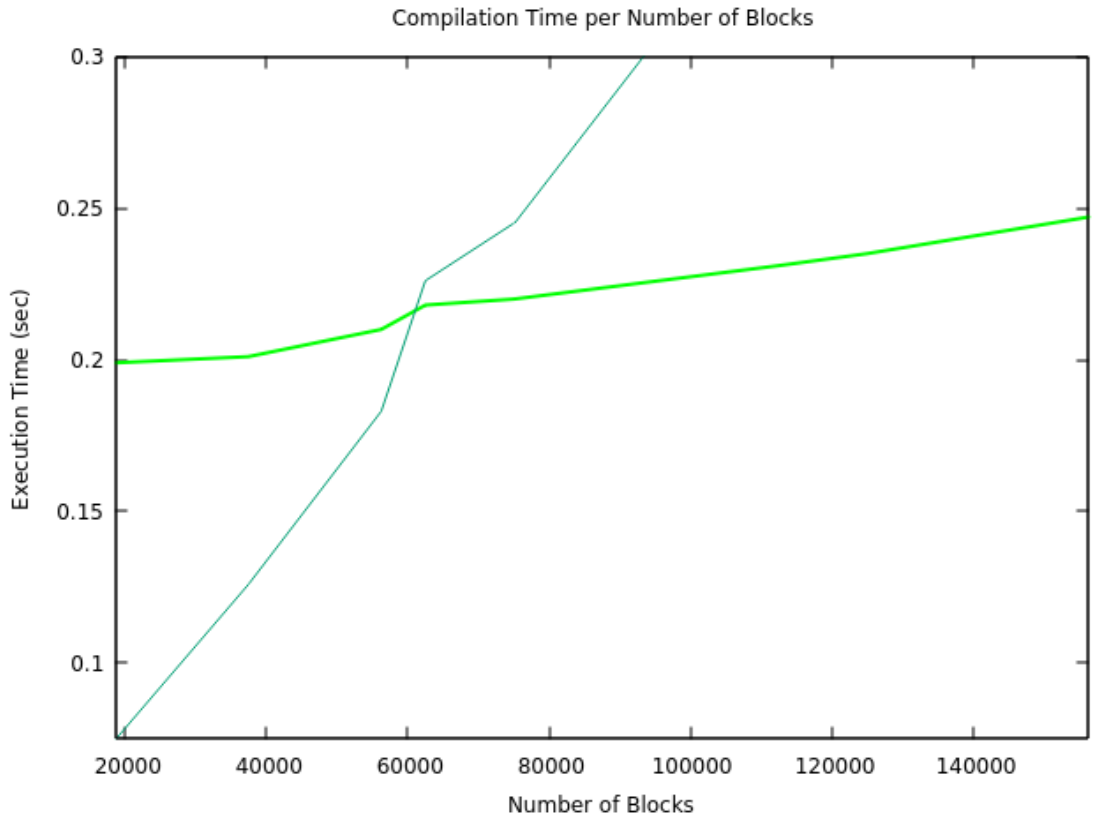
OpenMP: Execution time with default schedule

OpenMP vs Sequential



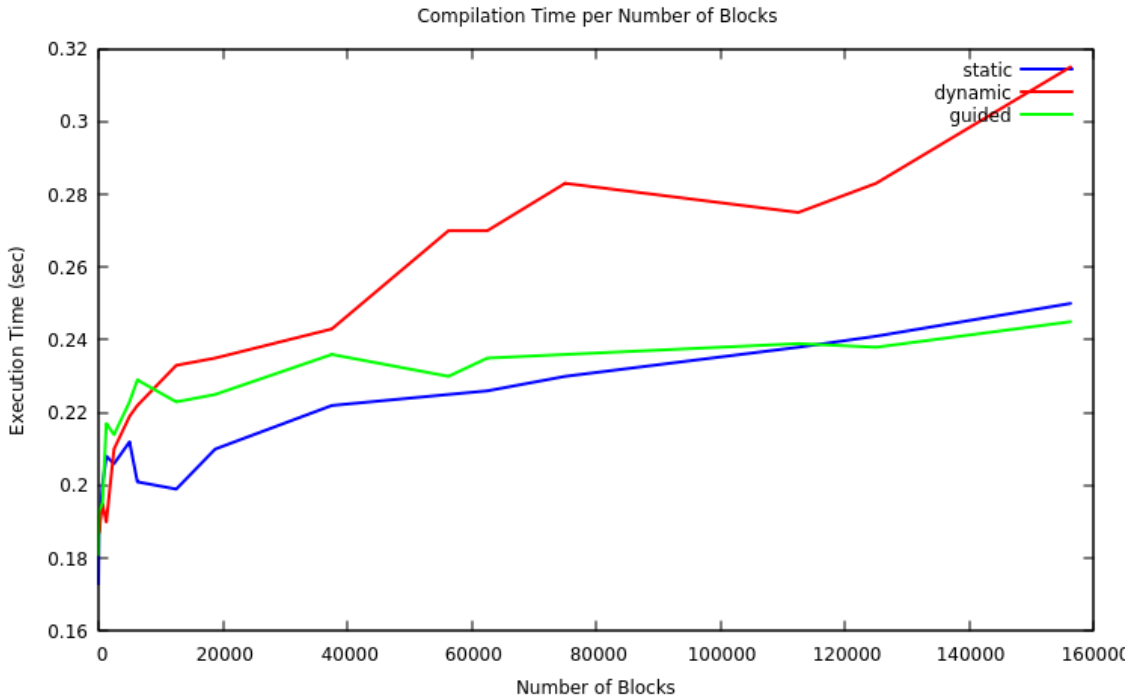
OpenMP: Execution time with default schedule

OpenMP vs Sequential



OpenMP: Scheduling Comparison

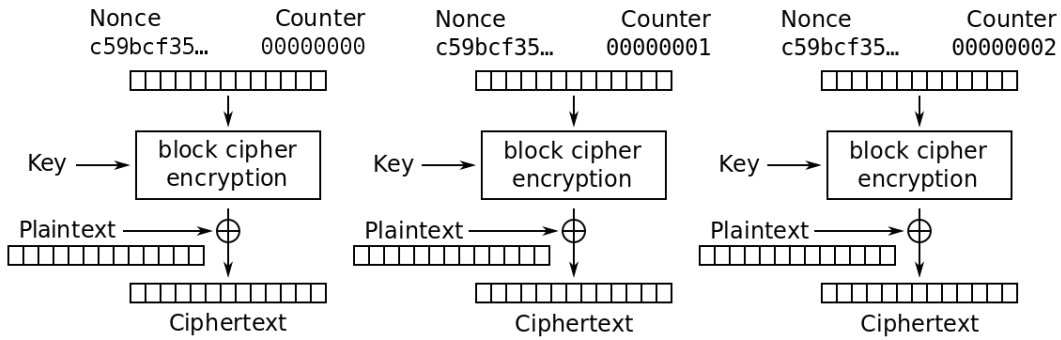
OpenMP: Scheduling Comparison



- workload equal per iteration (preferered static and guided)

MPI: What should we distribute?

AES CTR

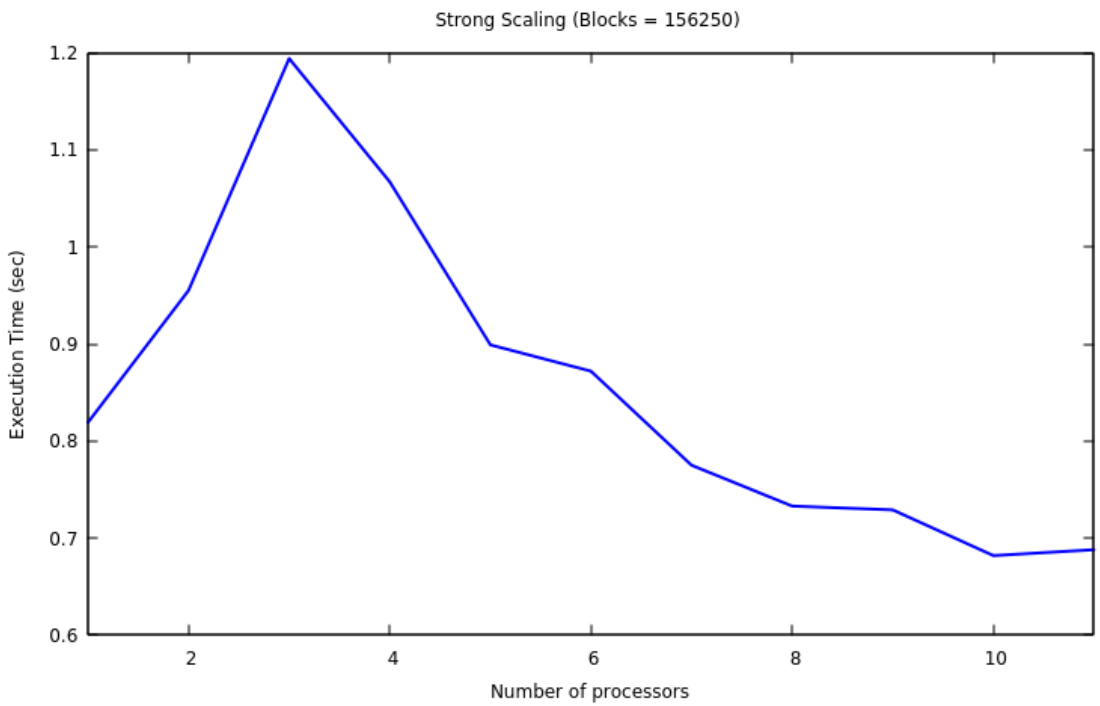


Counter (CTR) mode encryption

- We **cannot** distribute the plain text blocks.
- Idea: a master node assign IVs to slave nodes.
- Xor and recomposition on master node.
- Assumption: every node knows the private key.

Scaling analysis

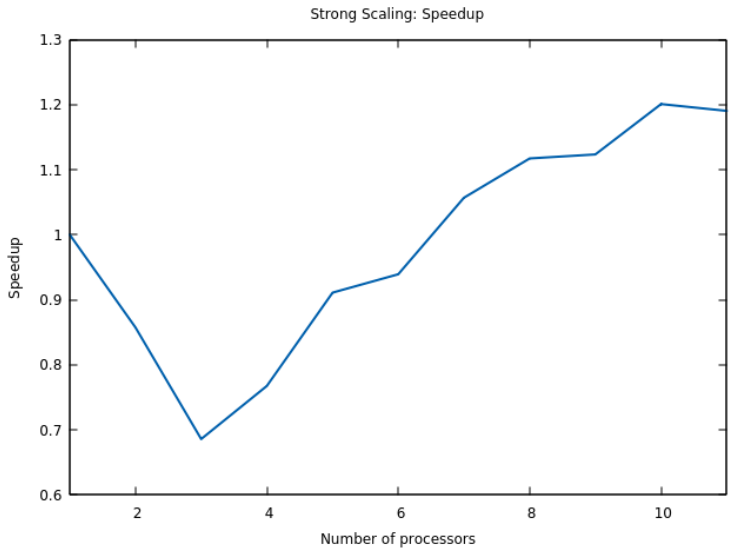
MPI: Strong Scaling

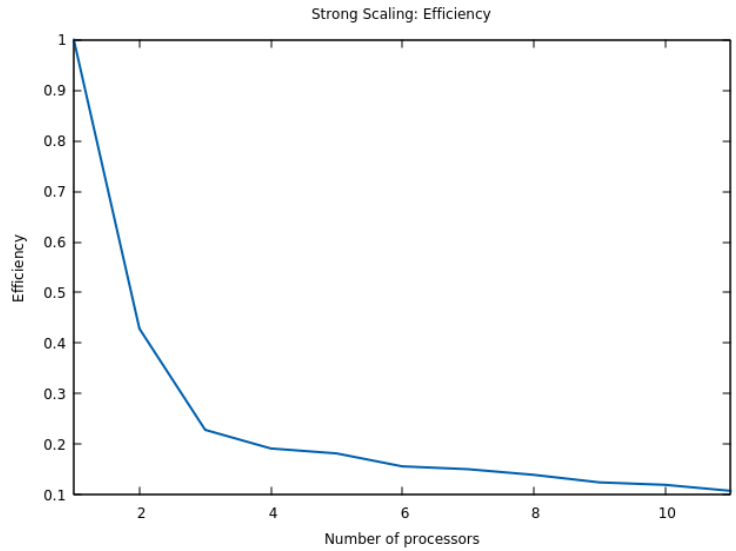


- With few node: "heavy" computation and communication costs.
- With many node: communication costs partially "absorbed" by small computation cost.

Scaling analysis

MPI: Strong Scaling (Speedup and Efficiency)

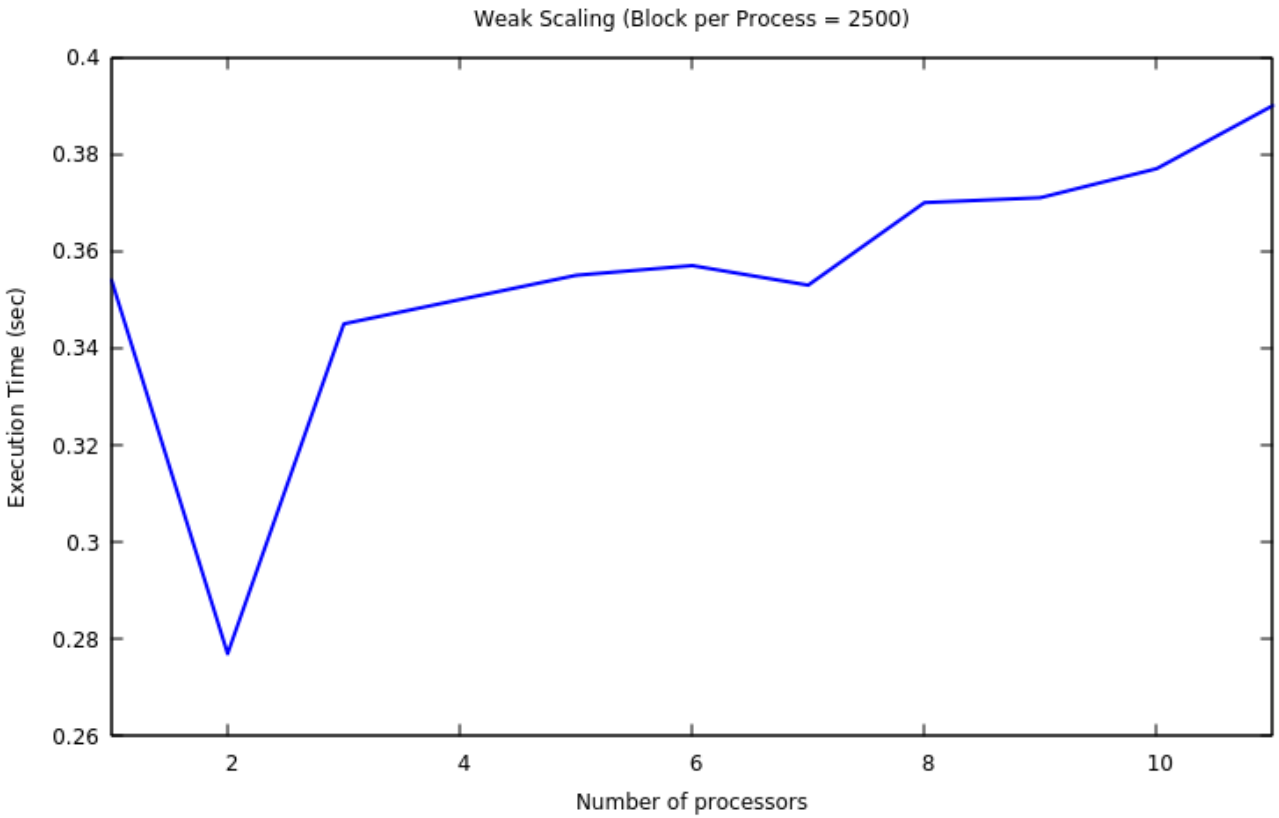




- Speedup increase slowly.
- Efficiency describe "how much" the speedup is proportional to the number of processors.

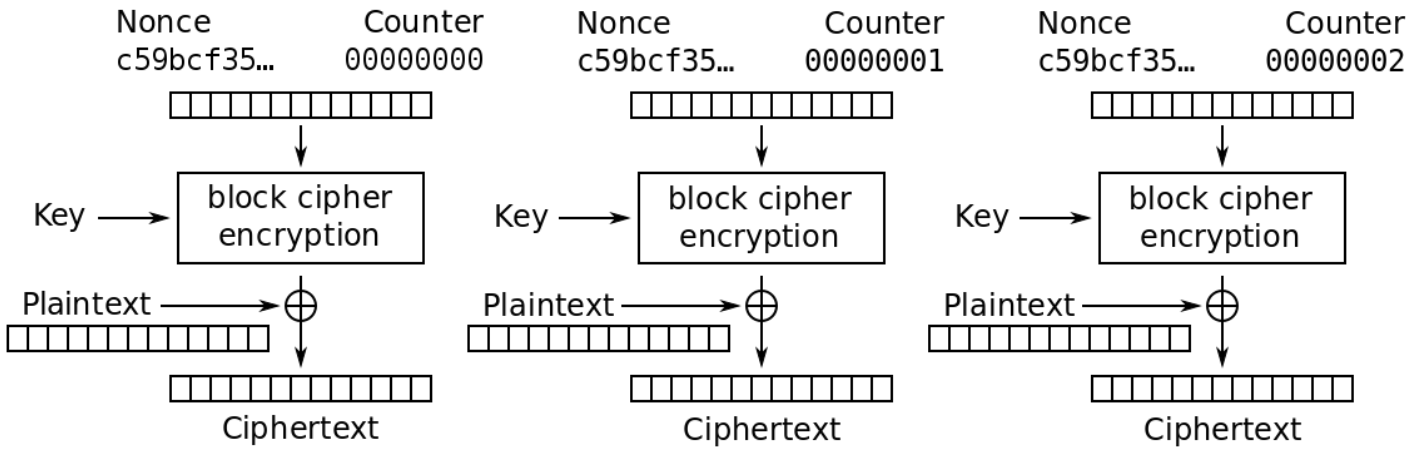
Scaling analysis

MPI: Weak Scaling



MPMPI: as before!

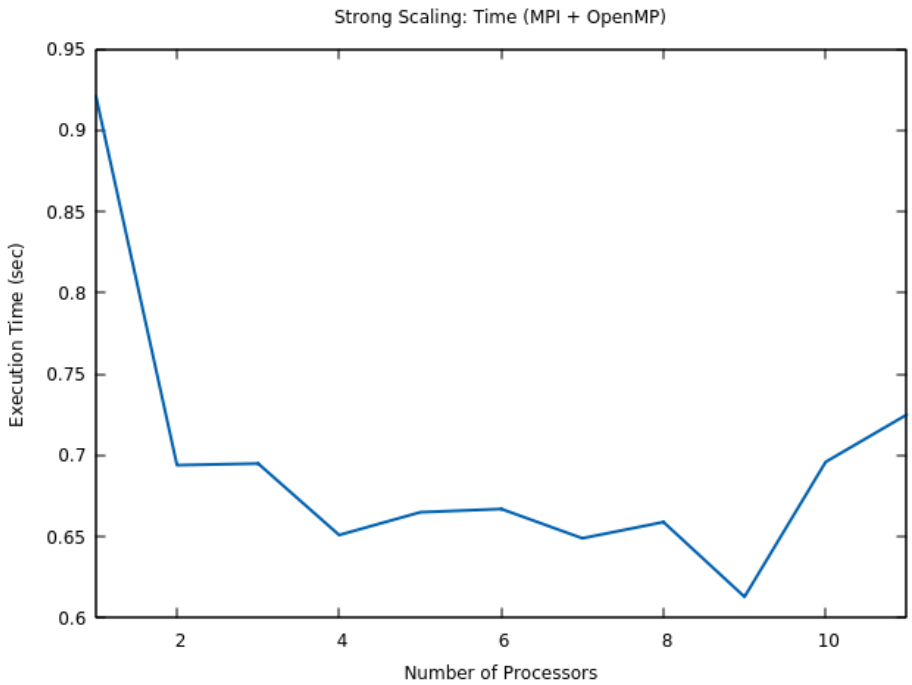
AES CTR



Counter (CTR) mode encryption

Scaling analysis

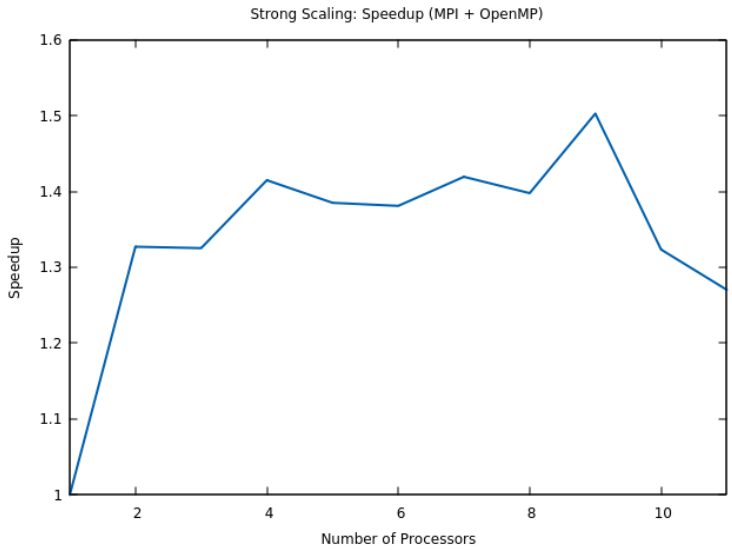
MPMPI: Strong Scaling

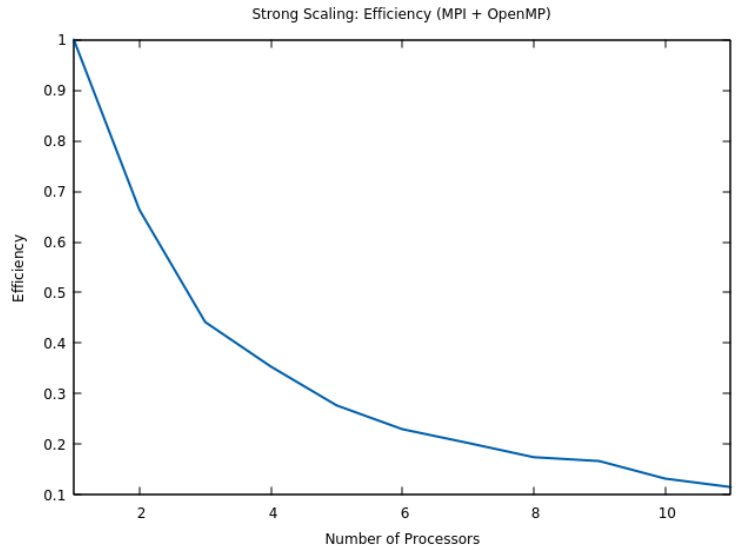


- Better than before!
- Parallelism helps since each node has a lot of blocks to encrypt.

Scaling analysis

MPMPI: Strong Scaling (Speedup and Efficiency)

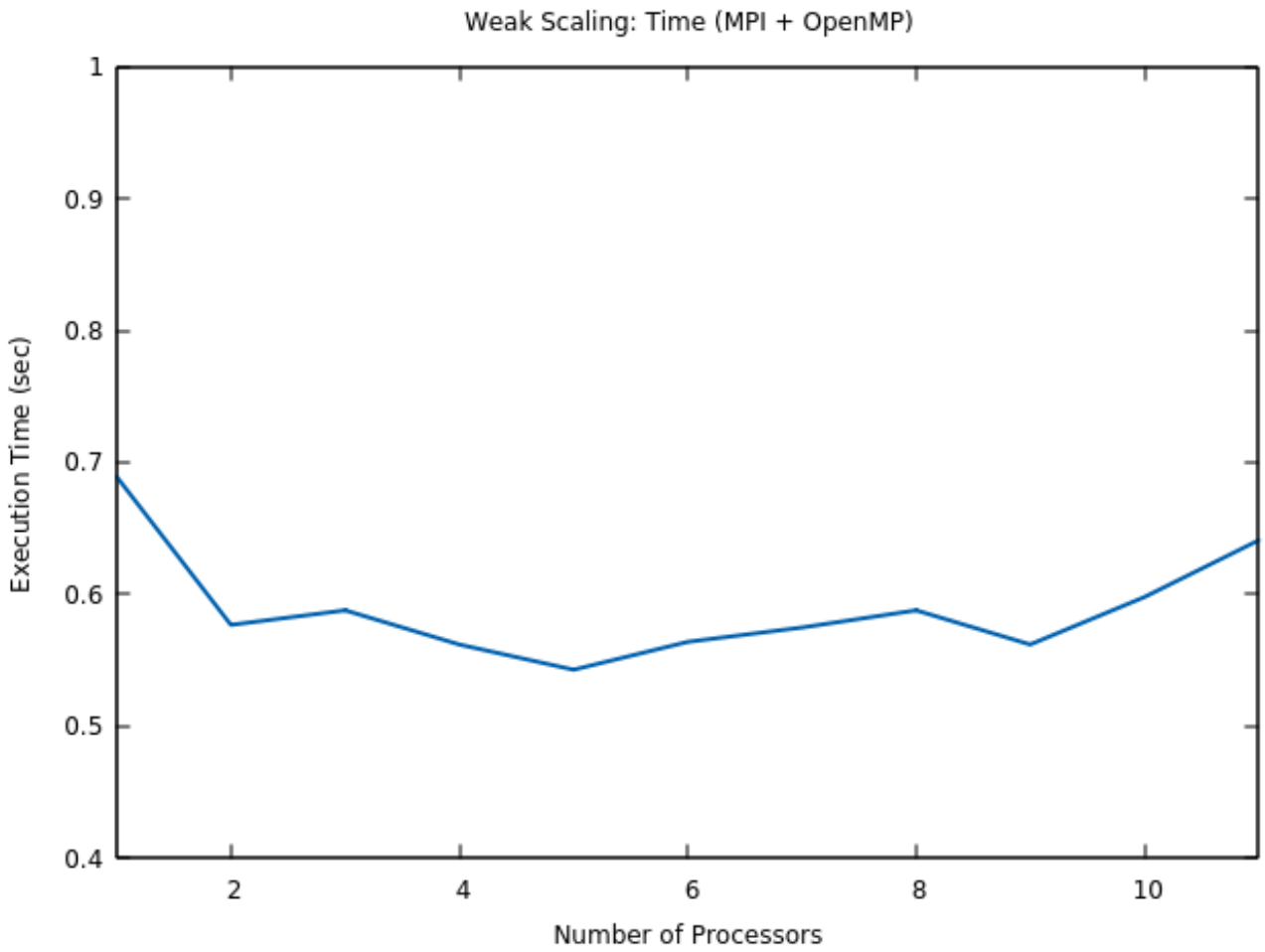




- Speedup increase more than MPI version.
- Efficiency more "proportional" than MPI version.

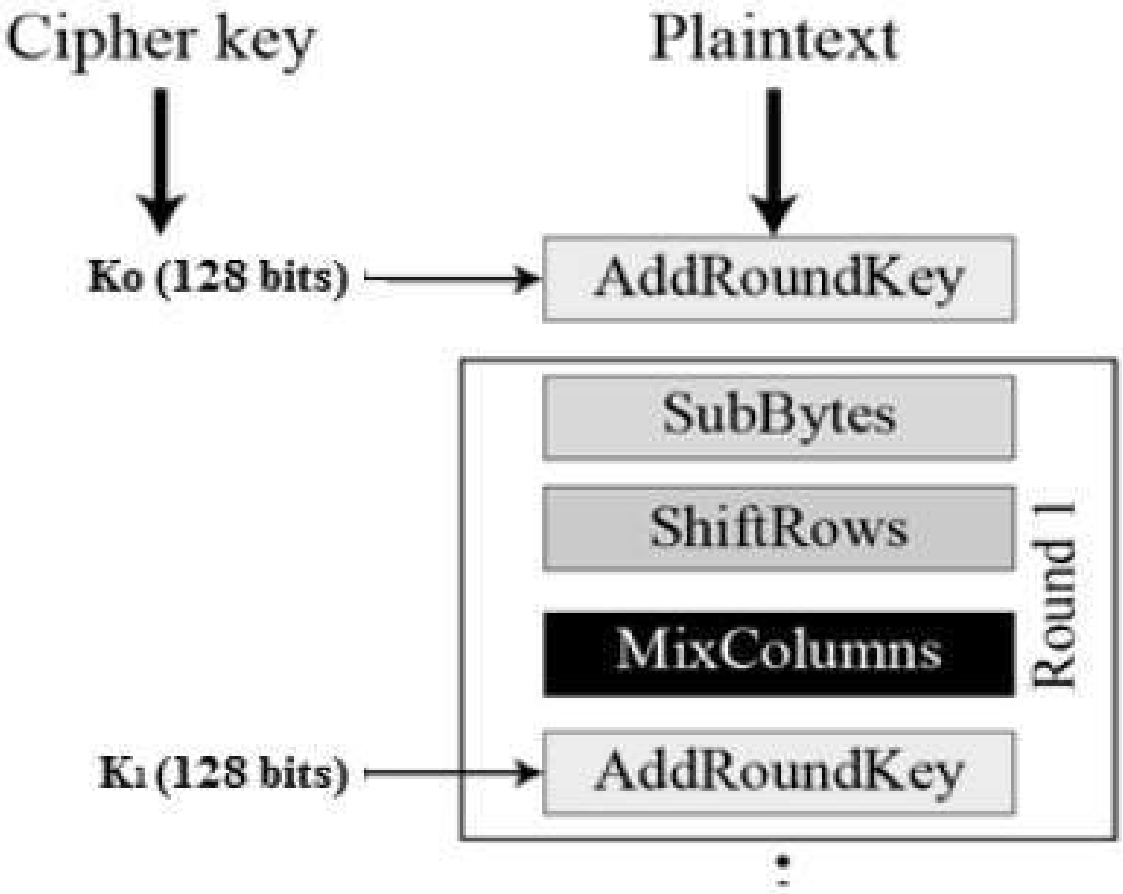
Scaling analysis

MPI: Weak Scaling



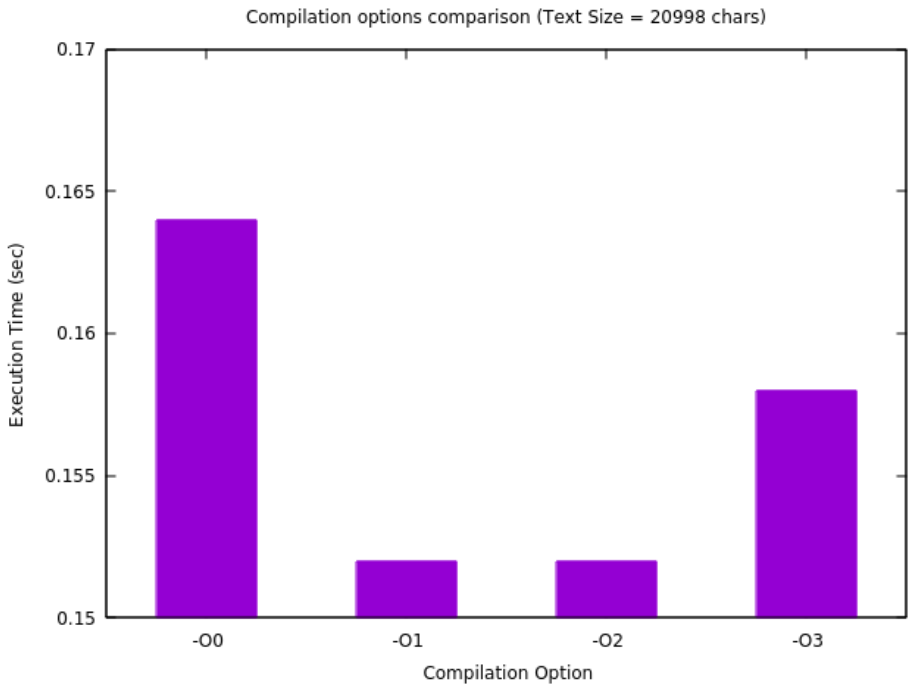
CUDA: everything on GPU!

AES on GPU



CUDA: Compilation Options

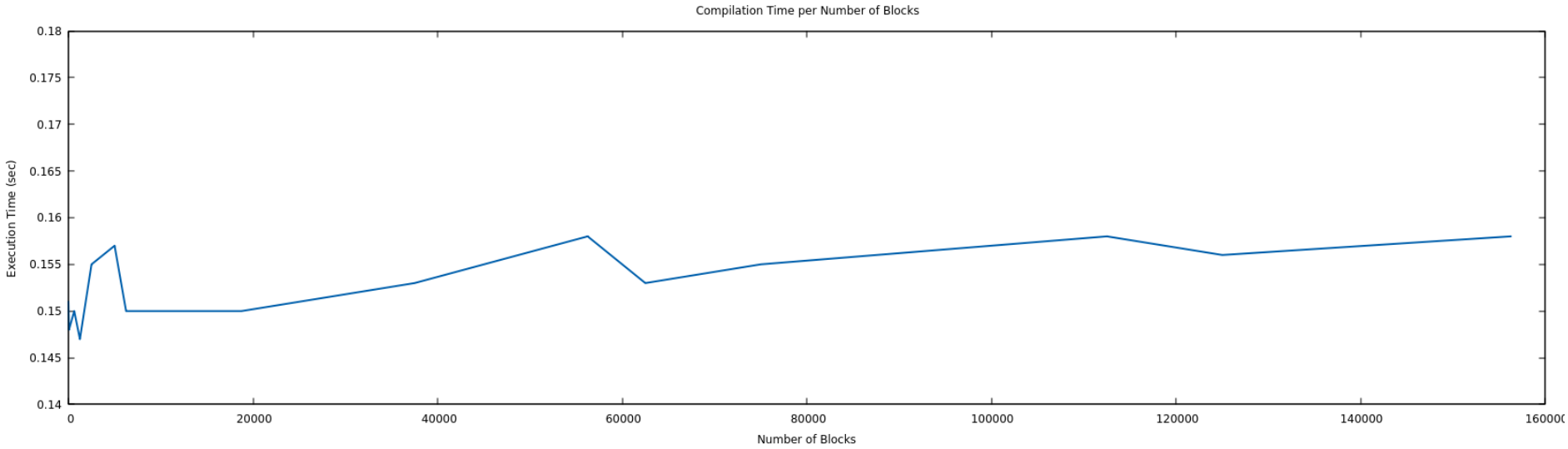
Compilation Options



- Execution time lower than Sequential version.
- O3 too aggressive.

CUDA: Compilation Options

Execution time per number of blocks



- Time curve grows very slowly.
- Time difference between the computation of 1 block and 156250 blocks of 0.007.

Conclusions: Reflections

- Sequential version is easy, doesn't require specific hardware but it's heavy for big files (execution time grows linearly with respect to the number of blocks).

Conclusions: Reflections

- Sequential version is easy, doesn't require specific hardware but it's heavy for big files (execution time grows linearly with respect to the number of blocks).
- OpenMP version is better than sequential after a certain number of blocks. It doesn't require specific hardware to run but it's not the best solution and for small files the sequential version is better.

Conclusions: Reflections

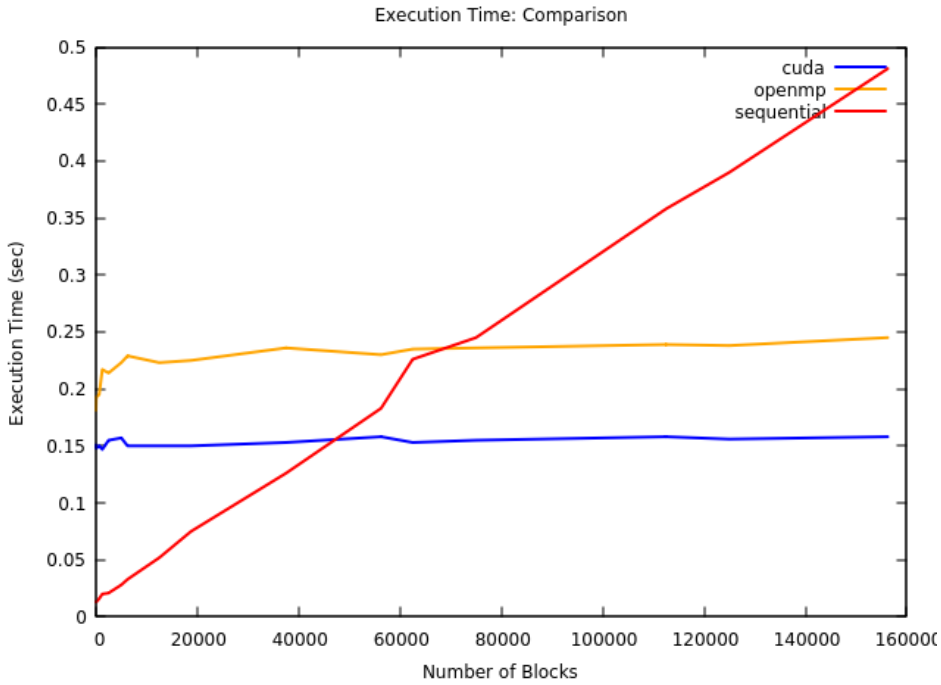
- Sequential version is easy, doesn't require specific hardware but it's heavy for big files (execution time grows linearly with respect to the number of blocks).
- OpenMP version is better than sequential after a certain number of blocks. It doesn't require specific hardware to run but it's not the best solution and for small files the sequential version is better.
- MPI and MPMPI (the hybrid version) are the worst versions in term of performances.

Conclusions: Reflections

- Sequential version is easy, doesn't require specific hardware but it's heavy for big files (execution time grows linearly with respect to the number of blocks).
- OpenMP version is better than sequential after a certain number of blocks. It doesn't require specific hardware to run but it's not the best solution and for small files the sequential version is better.
- MPI and MPMPI (the hybrid version) are the worst versions in term of performances.
- CUDA version is the best version in term of performances (after a certain number of blocks). It requires specific hardware to run (NVidia GPU) and for small file the sequential version is better. For medium and big file we get a huge increment in performances.

Conclusions: Final Comparison

Comparison between Sequential, OpenMP and CUDA



- Sequential good for small file (texts, small images etc).
- CUDA is the best options for medium and big file (raw, high quality audio and images, disk images etc).

Thanks for your attention!

Questions?