KU LEUVEN

Homomorphic string search with constant multiplicative depth

Charlotte Bonte

Ilia Iliashenko

imec-COSIC, KU Leuven, Belgium

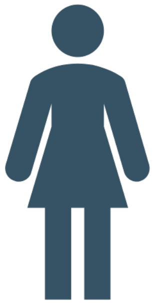
ACM Cloud Computing Security Workshop

9 November 2020

Secure search scenarios

Secure search scenarios

Client

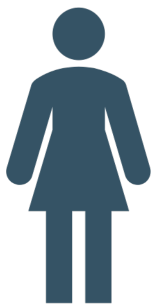


Server

Secure search scenarios

Secure search scenarios

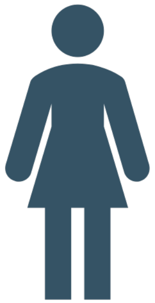
Client



Server

Secure search scenarios

Client



Server

Secure search scenarios

Client

Server

Secure search scenarios

Client

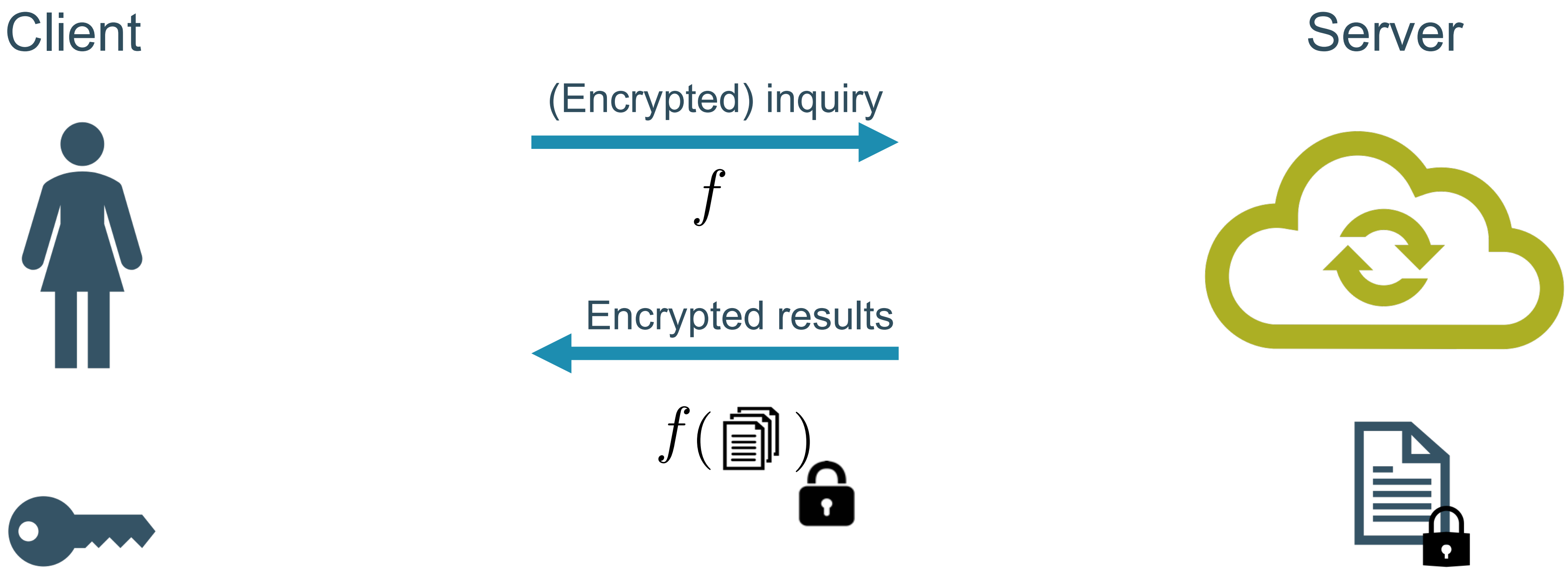
Server

Secure search scenarios

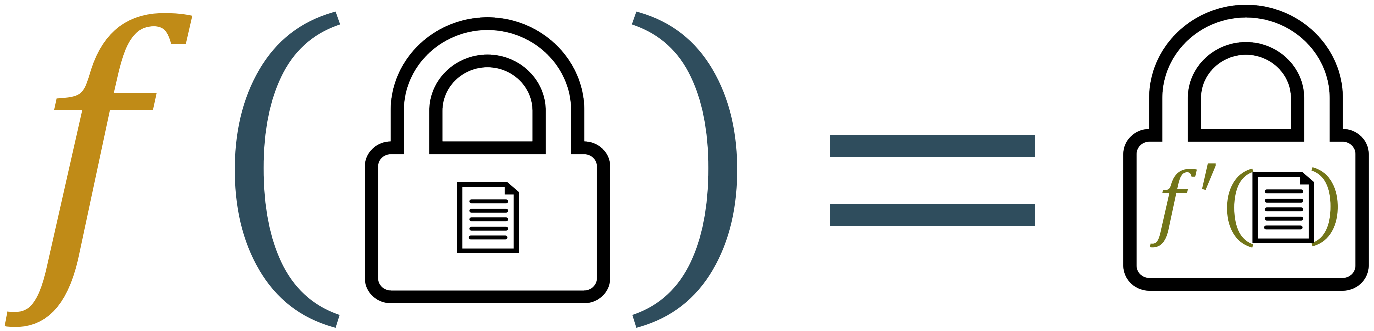
Client

Server

How to work with encrypted data in the cloud?



Homomorphic encryption (HE)

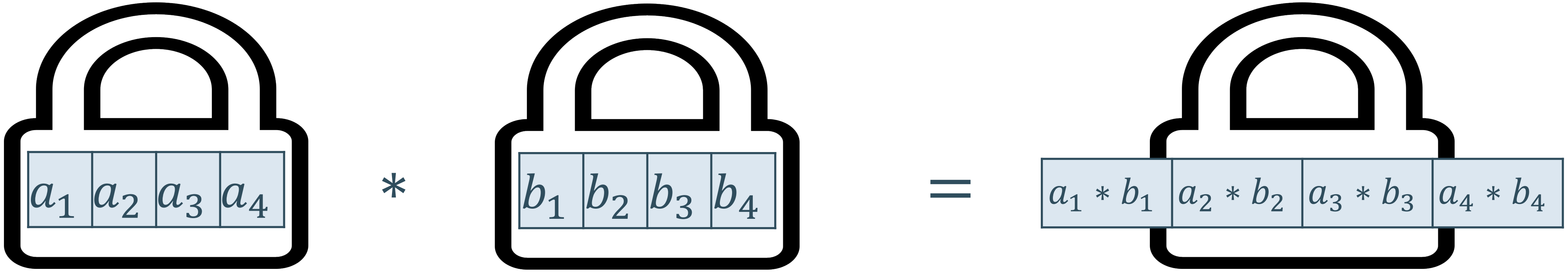


+ Low communication complexity (unlike MPC).
+ Non-interactiveness of computation (unlike MPC).
+ Universal functionality (unlike PIR, ORAM or PSI).
+ No data leakage (unlike SSE).

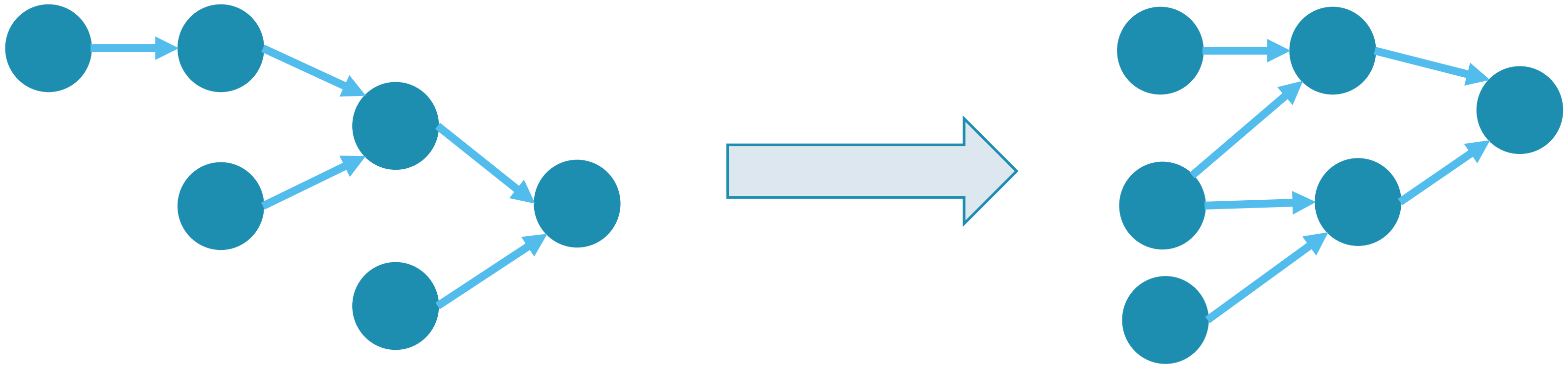
- Running time and memory overhead.

Homomorphic encryption optimisations

- SIMD Packing



- Reduction of multiplicative depth



String search

Text	T	T	G	A	G	A	G	A	C	C
Pattern	G	A	G	A						

0	0	1	0	1	0	0	0	0	0

String search

Text

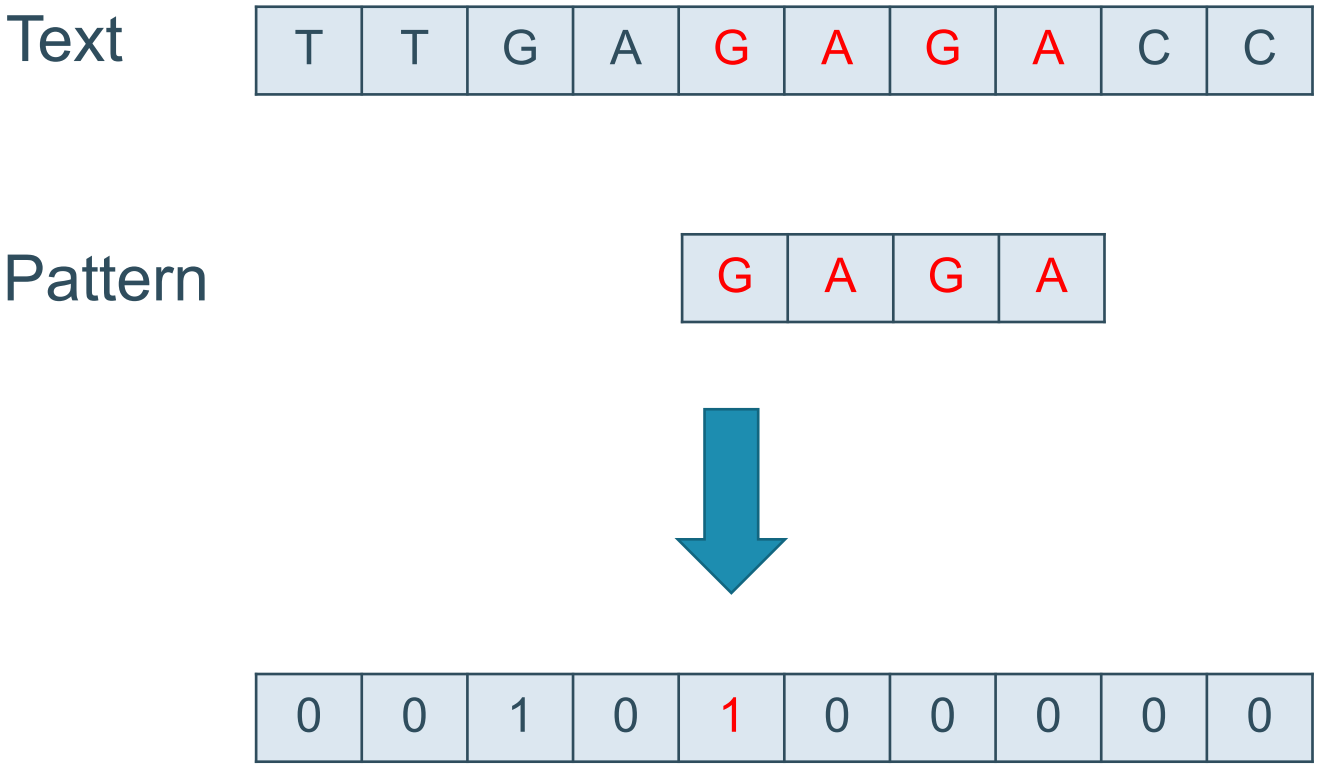
T	T	G	A	G	A	G	A	C	C

Pattern

G	A	G	A

0	0	1	0	1	0	0	0	0	0

String search



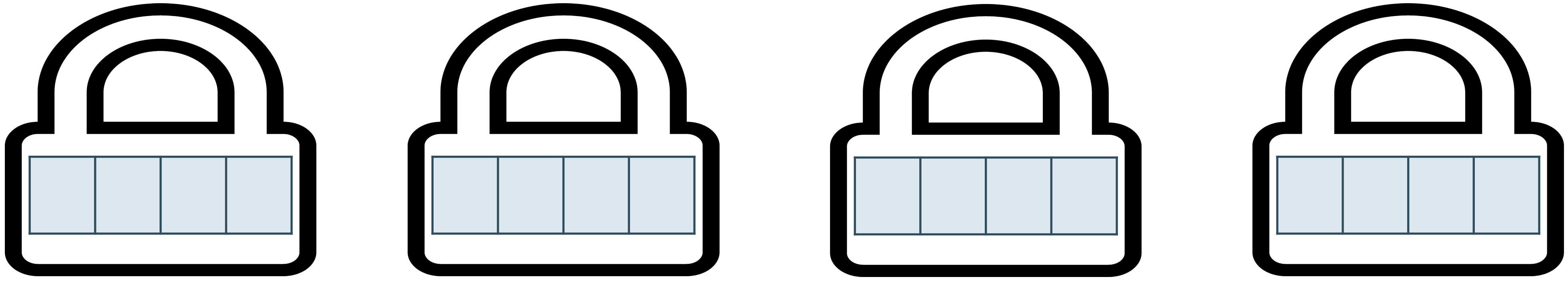
Our contribution

General framework for string search in HE with SIMD packing

- Preprocessing
  - Encoding of patterns and large texts into HE plaintexts
- Processing
  - String search with a multiplicative depth independent of the input length (up to 12 times faster than the state of the art)
- Postprocessing
  - Compression of string search results (reduces the communication complexity by a large constant factor)

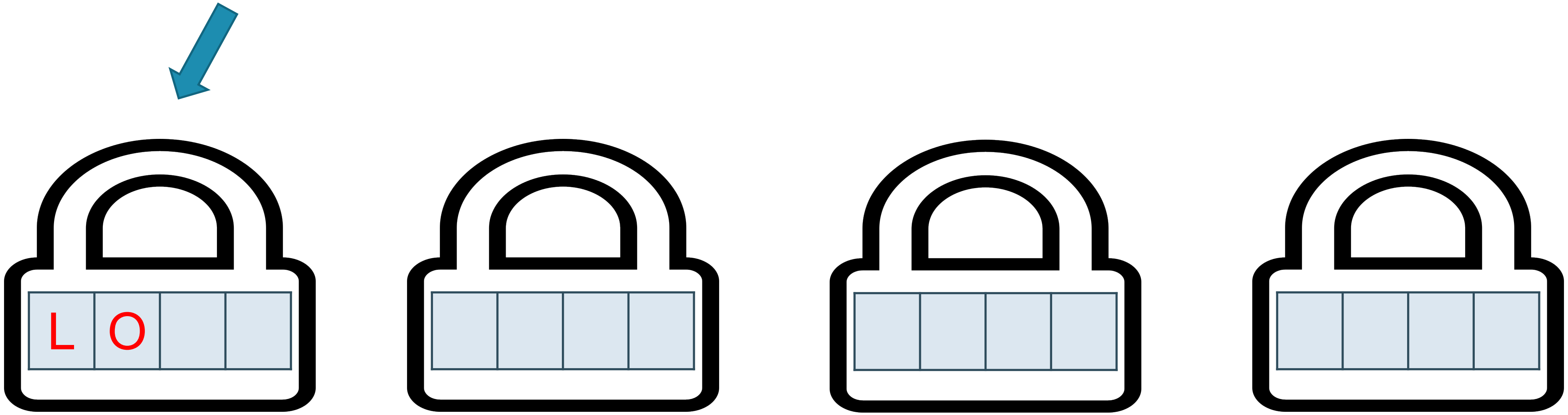
Preprocessing: how to encrypt the text?

Text	L	O	R	E	M	_	I	P	S	U	M
Pattern	?	?									



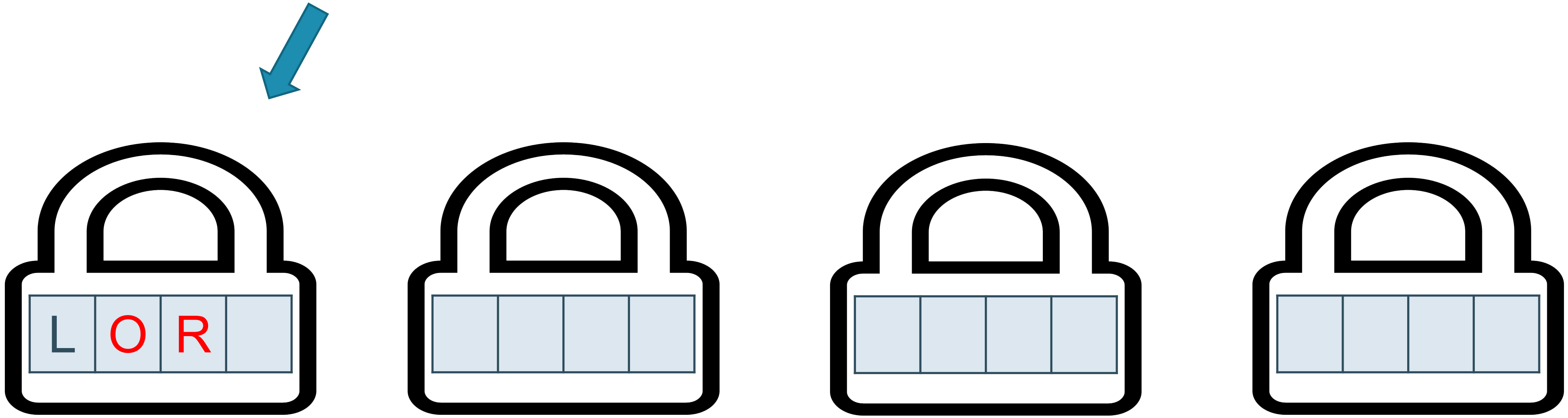
Preprocessing: how to encrypt the text?

Text	L	O	R	E	M	_	I	P	S	U	M
Pattern	?	?									



Preprocessing: how to encrypt the text?

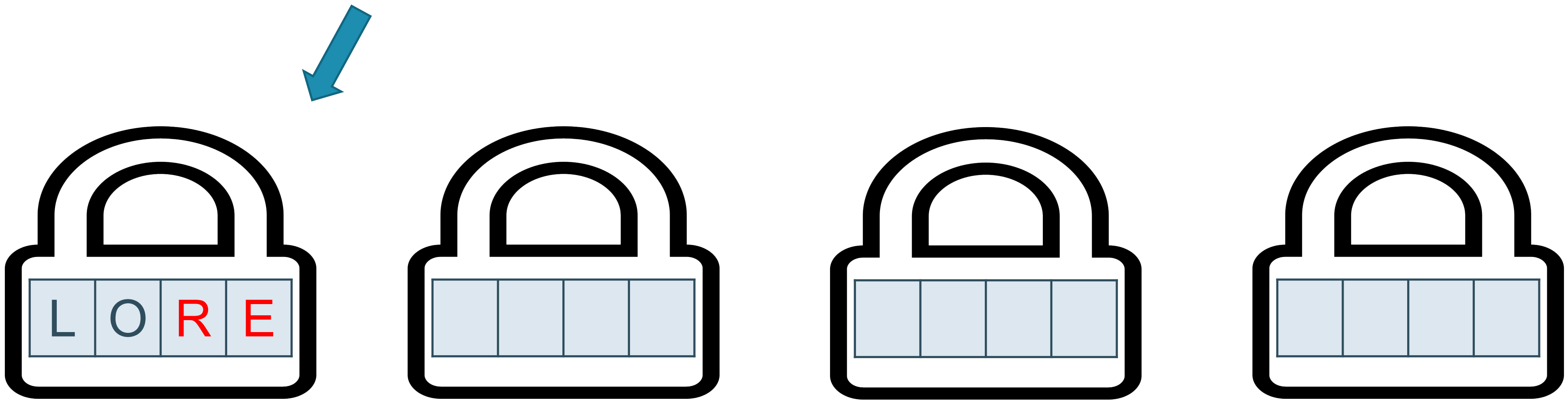
Text	L	O	R	E	M	_	I	P	S	U	M
Pattern	?	?									



Preprocessing: how to encrypt the text?

Text	L	O	R	E	M	_	I	P	S	U	M

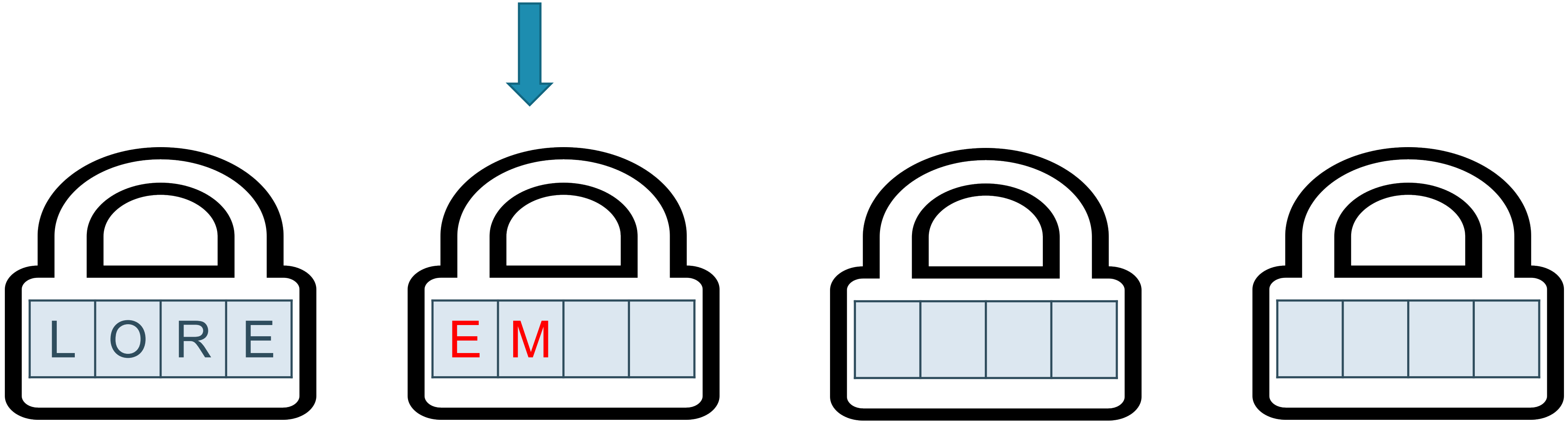
Pattern	?	?



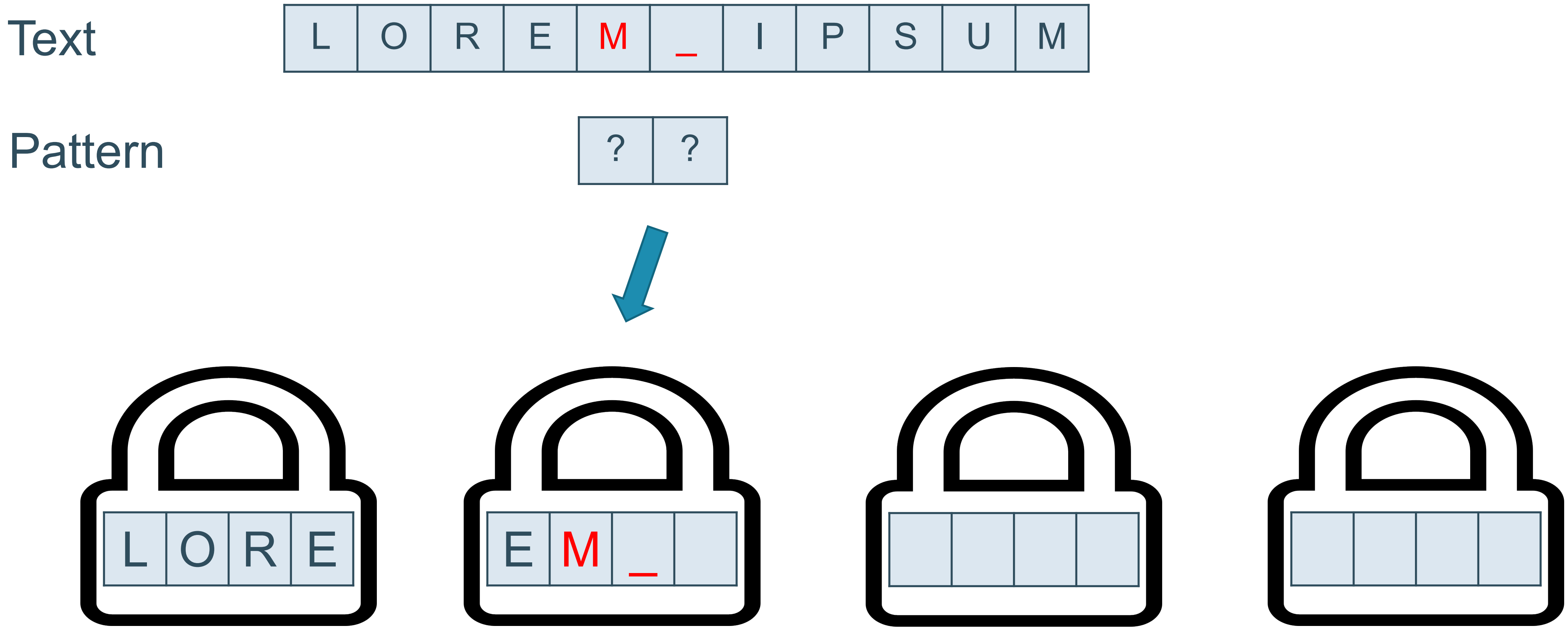
Preprocessing: how to encrypt the text?

Text	L	O	R	E	M	_	I	P	S	U	M

Pattern	?	?



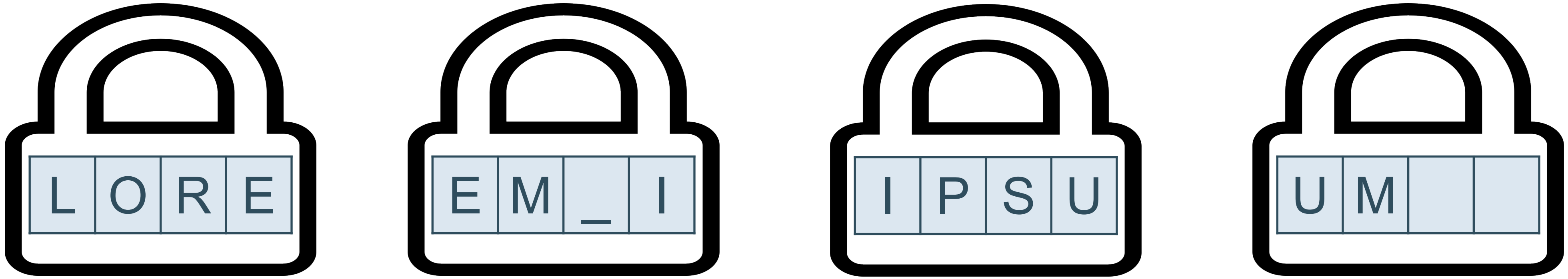
Preprocessing: how to encrypt the text?



Preprocessing: how to encrypt the text?

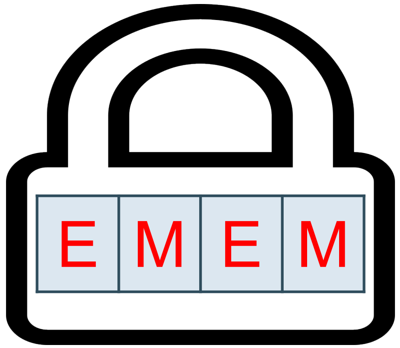
Text	L	O	R	E	M	_	I	P	S	U	M
Pattern	?	?									

$r = \left\lceil \frac{|text|-|pattern|+1}{|slots|-|pattern|+1} \right\rceil$ ciphertexts are needed



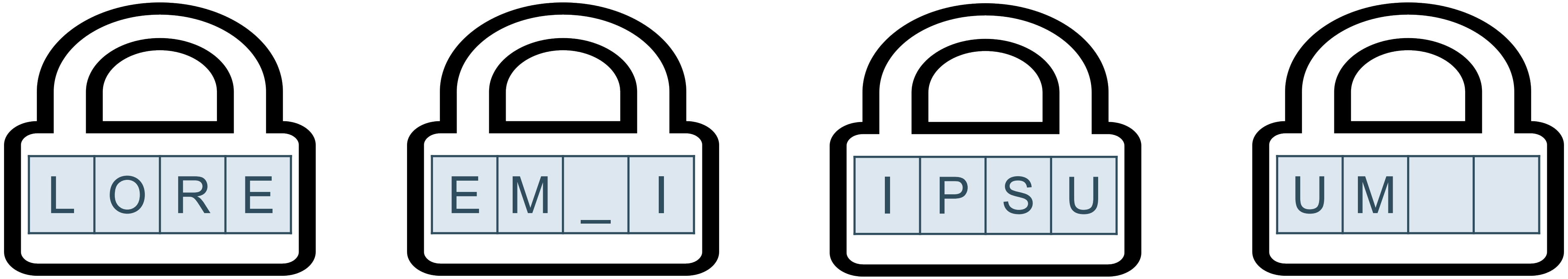
Preprocessing: how to encrypt the pattern?

Text	L	O	R	E	M	_	I	P	S	U	M
Pattern	E	M									



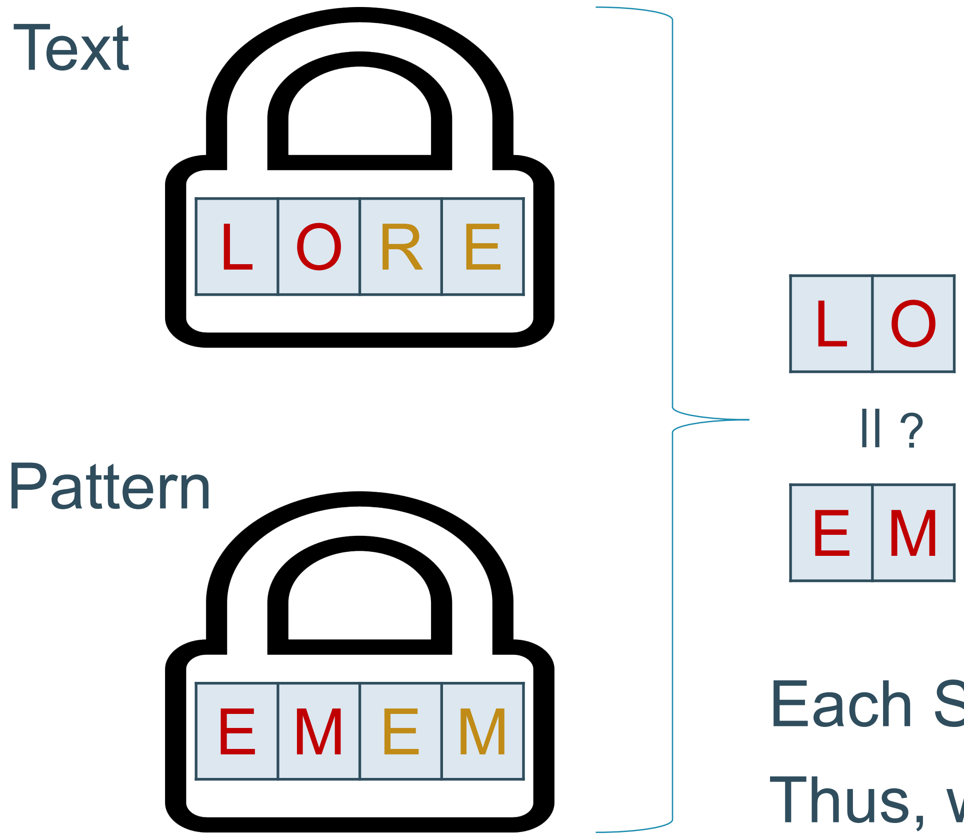
Processing: how to match strings

Text



Pattern

Processing: how to match strings



Each SIMD slot is isomorphic to $\mathbb{F}_q$.

Thus, we need to compare vectors in $\mathbb{F}_q^{|pattern|}$.

Processing: exact matching

Text	t_1	t_2	…	t_ℓ	$= \boldsymbol{t} \in \mathbb{F}_{\boldsymbol{q}}^{\ell}$
		‖ ?			
Pattern	p_1	p_2	…	p_ℓ	$= \boldsymbol{p} \in \mathbb{F}_{\boldsymbol{q}}^{\ell}$

$$EQ(\boldsymbol{t}, \boldsymbol{p}) = \prod_{i=1}^{\ell} 1 - (t_i - p_i)^{q-1} = \begin{cases} 0 \ if \ \boldsymbol{t} \neq \boldsymbol{p} \\ 1 \ if \ \boldsymbol{t} = \boldsymbol{p} \end{cases}$$

Multiplicative depth depends on the pattern length ℓ.

Processing: randomized matching

Text	t_1	t_2	…	t_ℓ	$= \boldsymbol{t} \in \mathbb{F}_{\boldsymbol{q}}^{\ell}$
		|| ?			
Pattern	p_1	p_2	…	p_ℓ	$= \boldsymbol{p} \in \mathbb{F}_{\boldsymbol{q}}^{\ell}$

$$EQ^r(\boldsymbol{t},\boldsymbol{p}) = 1 - \left(\sum_{i=1}^{\ell} r_i(t_i - p_i)\right)^{q-1} = \begin{cases} 0 \text{ if } \boldsymbol{t} \neq \boldsymbol{p} & \text{with probability } 1 - 1/q \\ 1 \text{ if } \boldsymbol{t} = \boldsymbol{p} & \text{always} \end{cases}$$

r_i is a uniformly random element of $\mathbb{F}_q$.

Processing: randomized matching

$$EQ^r(\boldsymbol{t},\boldsymbol{p}) = 1 - \left(\sum_{i=1}^{\ell} r_i(t_i - p_i)\right)^{q-1} = \begin{cases} 0 \; if \; \boldsymbol{t} \neq \boldsymbol{p} & \text{with probability } 1 - 1/q \\ 1 \; if \; \boldsymbol{t} = \boldsymbol{p} & \text{always} \end{cases}$$

q must be large, but $q - 1$ must be small

Processing: randomized matching

$$EQ^r(\boldsymbol{t},\boldsymbol{p}) = 1 - \left(\sum_{i=1}^{\ell} r_i(t_i - p_i)\right)^{q-1} = \begin{cases} 0 \; if \; \boldsymbol{t} \neq \boldsymbol{p} & \text{with probability } 1 - 1/q \\ 1 \; if \; \boldsymbol{t} = \boldsymbol{p} & \text{always} \end{cases}$$

q must be large, ~~but $q-1$ must be small~~

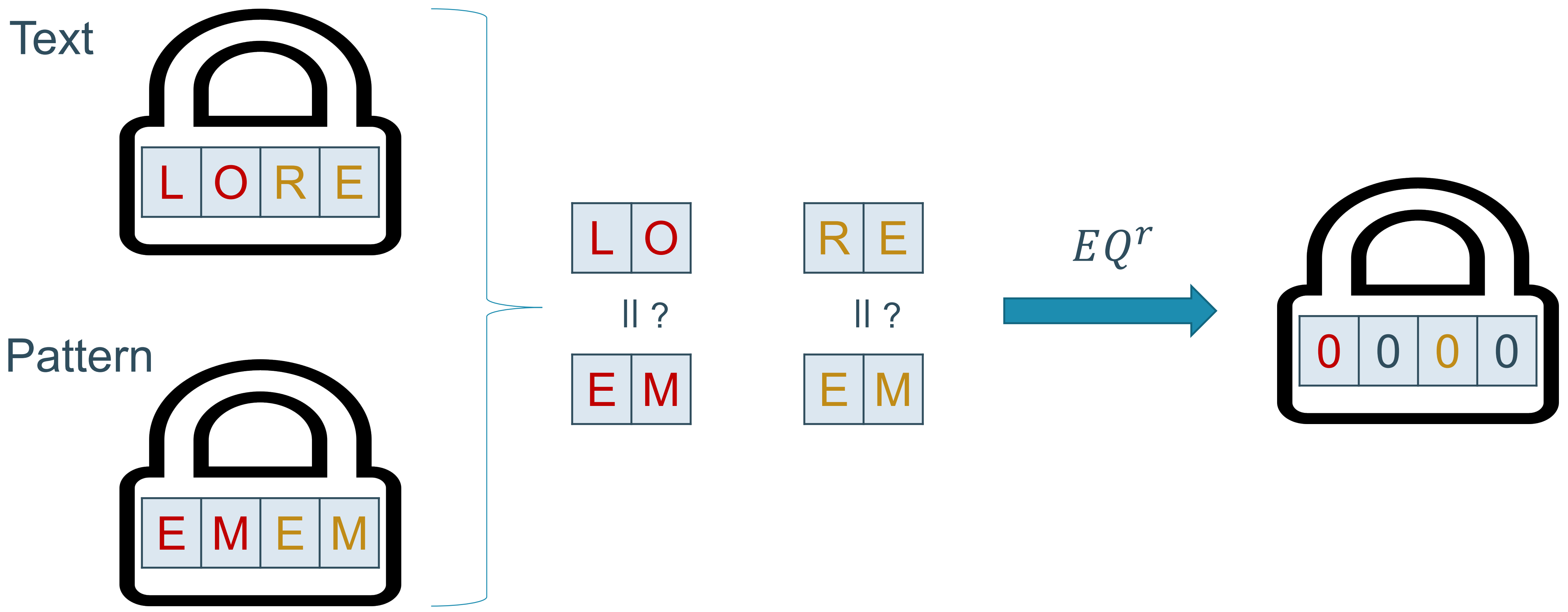
If $q = p^k$, then

$$a^{q-1} = a^{p^k-1} = a^{(p-1)(p^{k-1}+p^{k-2}+\cdots p+1)} = (a^{p-1})^{p^{k-1}} \cdot (a^{p-1})^{p^{k-2}} \cdots a^{p-1}$$

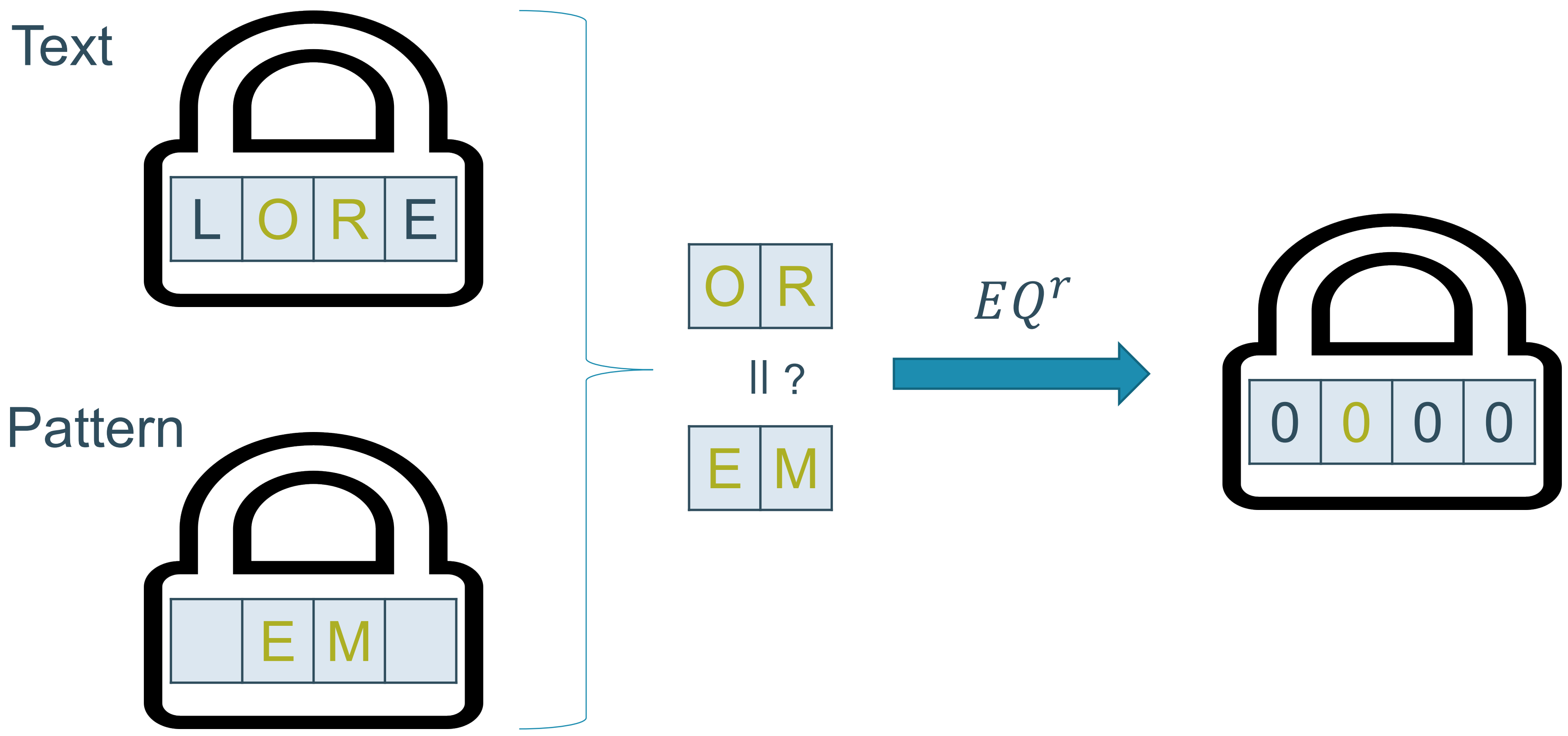
Exponentiation a^p (the Frobenius map) is free of homomorphic multiplication.

Mult. depth: $\lceil \log_2(p-1) \rceil + \lceil \log_2 k \rceil$, **independent of the pattern length** ℓ.

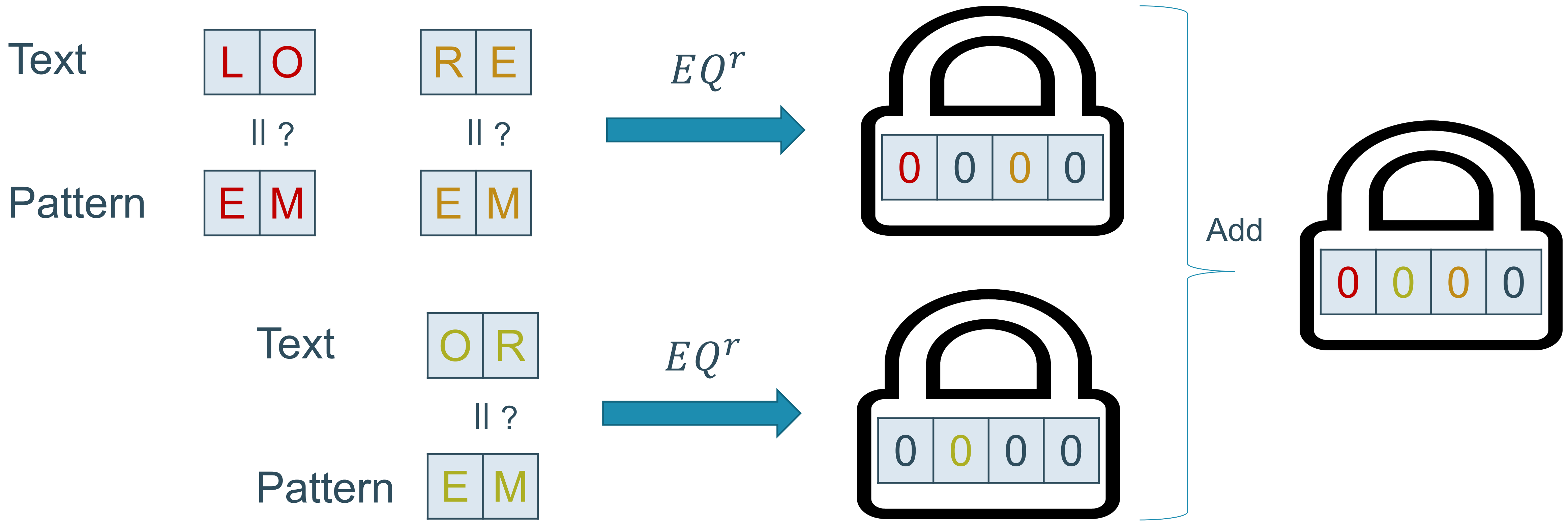
Processing: how to match strings



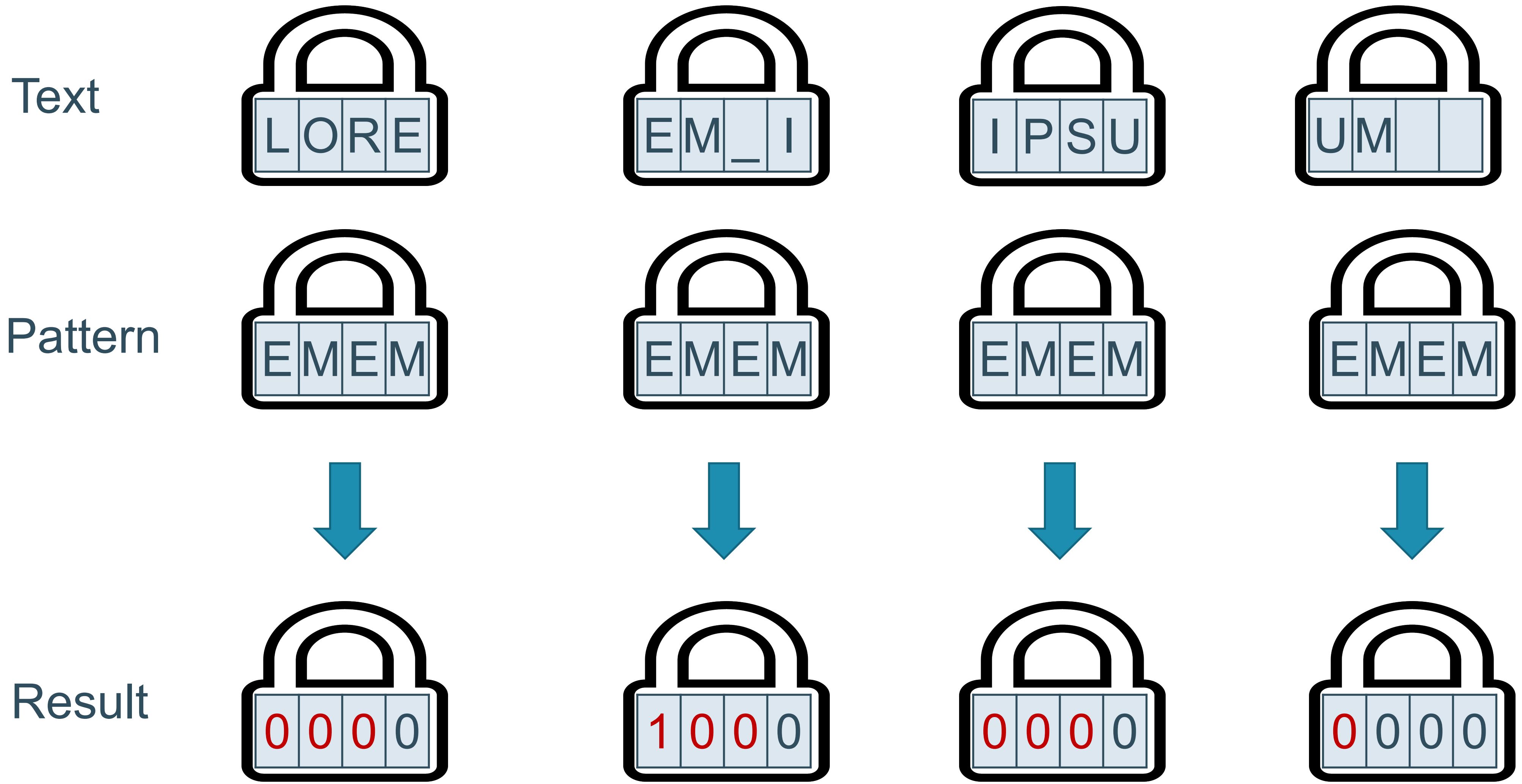
Processing: how to match strings



Processing: how to match strings

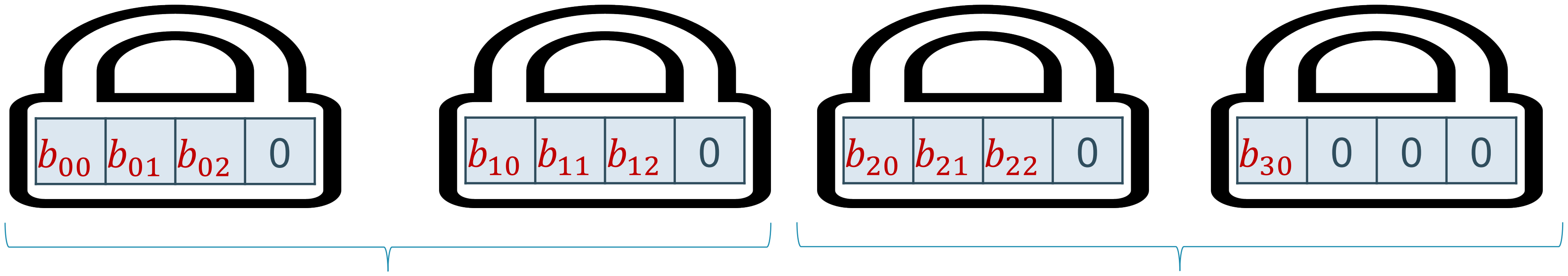


Processing: how to match strings



Postprocessing: compression

Result



Each SIMD slot is isomorphic to $\mathbb{F}_q \cong \mathbb{F}_p[X]/(f(X))$, where $\deg f(X) = d$.
Thus, we can encode $d \cdot \lfloor \log_2 p \rfloor$ bits in one slot.

Mult. by 2 + Add

Mult. by 2 + Add

$b_{00} + 2b_{10}$	$b_{01} + 2b_{11}$	$b_{02} + 2b_{12}$	0

$b_{20} + 2b_{30}$	b_{21}	b_{22}	0

Mult. by X
+ Add

$(b_{00}+2b_{10})$ $+ (b_{20} + 2b_{30})X$	$(b_{01}+2b_{11})$ $+ b_{21}X$	$(b_{02}+2b_{12})$ $+ b_{22}X$	0

Results

Text length	#ciphertexts with text	Output size, MB	Time, min	Failure probability
10000	10	0.35	52	2^{-20}
100000	97	1.69	506	2^{-17}
1000000	970	13.9	5060	2^{-14}

UTF-32 characters including wildcards are used.

The pattern length is 50.

Parallel computation takes 5 minutes in all three cases.

	Independent mult. depth	Compression	Pattern length	Failure prob. per substring	Time per bit, ms
Exact matching (Kim et al., 2017)	No	No	35 - 55	0	7.07 - 10.86
Randomized binary matching (Akavia et al., 2019)	No	No	64	2^{-1} - 2^{-80}	1.58 - 5.50
This work	Yes	Yes	1 - 100	2^{-34} - 2^{-65}	0.06 - 0.13

Thank you!

Time for Q&A!