Constant-weight PIR: Single-round Keyword PIR via Constant-weight Equality Operators

Rasoul Akhavan Mahdavi, Florian Kerschbaum

Usenix 2022, Boston, US

UNIVERSITY OF WATERLOO

CrySP
Cryptography, Security, and Privacy Research Group

Homomorphic Encryption

- Method for computation over encrypted data
- Circuits from basic operations: + and *

Homomorphic Encryption

- Method for computation over encrypted data
- Circuits from basic operations: + and *
- Multiplication is expensive
- Mult. Depth determines parameters

Operation	Time (μs)				Noise Growth
	$N = 2048$	$N = 4096$	$N = 8192$	$N = 16384$	
Addition	6	19	67	435	Additive
Plain Mult.*	12–135	30–529	105–2201	509–9647	Multiplicative
Multiplication	-	3823	15744	66908	Multiplicative
Substitution	-	768	4137	26047	Additive

Larger Parameters

UNIVERSITY OF WATERLOO

Homomorphic Encryption

- Method for computation over encrypted data
- Circuits from basic operations: + and *
- Multiplication is expensive
- Mult. Depth determines parameters

Operation	Time (μs)				Noise Growth
	$N = 2048$	$N = 4096$	$N = 8192$	$N = 16384$	
Addition	6	19	67	435	Additive
Plain Mult.*	12–135	30–529	105–2201	509–9647	Multiplicative
Multiplication	-	3823	15744	66908	Multiplicative
Substitution	-	768	4137	26047	Additive

Larger Parameters

1. Use Fewer Multiplications
2. Reduce the Multiplicative Depth

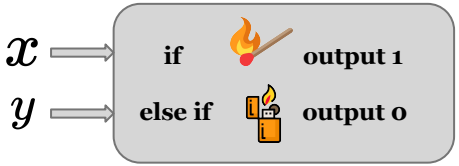
UNIVERSITY OF WATERLOO

Equality Operators using HE

- Definition

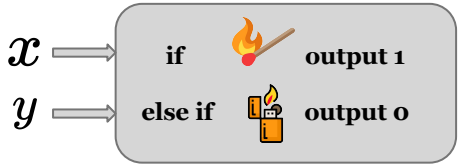
UNIVERSITY OF WATERLOO

Equality Operators using HE



- Definition
- Important building block
  - Private Information Retrieval , Private Set Intersection, …

Equality Operators using HE



- Definition:
- Important building block
  - Private Information Retrieval , Private Set Intersection, …
- Existing equality operators

$$D = \{0, 1\}^{\ell}$$

Plain

$$f_{FP}(x, y) = \prod_{y[i]=0} (1 - x[i]) \prod_{y[i]=1} x[i]$$

Arithmetic

$$f_{AF}(x, y) = \prod_{i=0}^{\ell-1} \left(1 + (x[i] - y[i])^2\right)$$

Equality Operators using HE

- Multiplicative depth depends on the element size

Arithmetic

$$f_{AF}(x, y) = \prod_{i=0}^{\ell-1} \left(1 + (x[i] - y[i])^2\right)$$

# of multiplications: 2ℓ

Multiplicative depth: $\log_2 \ell + 1$

- Limits scalability of equality using HE

Equality Operators using HE

- Multiplicative depth depends on the element size

Arithmetic

$$f_{AF}(x, y) = \prod_{i=0}^{\ell-1} \left(1 + (x[i] - y[i])^2\right)$$

# of multiplications: 2ℓ

Multiplicative depth: $\log_2 \ell + 1$

- Limits scalability of equality using HE

We need a better approach

UNIVERSITY OF WATERLOO

Constant-weight Code

Constant-weight codewords

Exactly k bits set to one

m=6, k=2

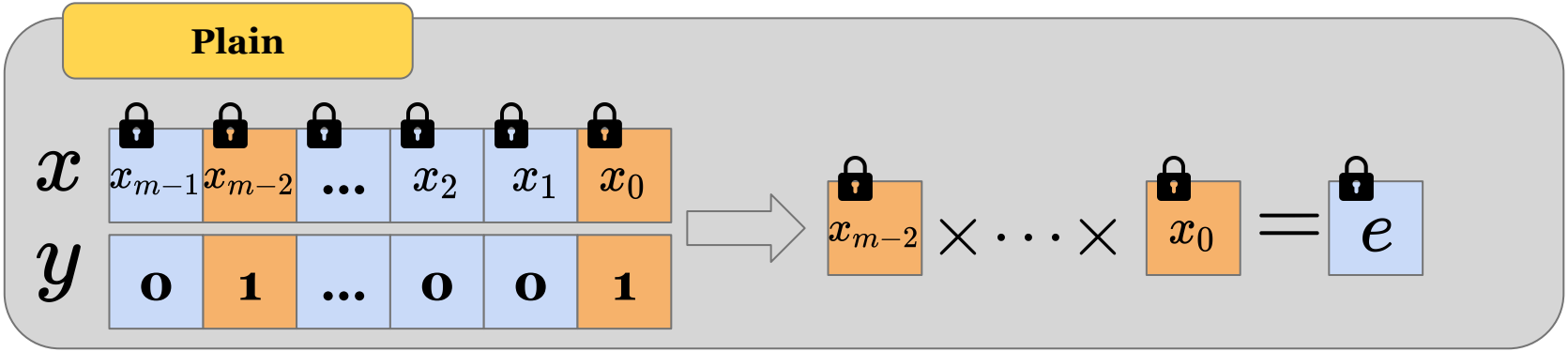
000011
000101
000110
001001
001010
001100
.
.
.

Equality Operator for Constant-weight Codewords

Plain

x	x_{m-1}	x_{m-2}	...	x_2	x_1	x_0
y	0	1	...	0	0	1

Equality Operator for Constant-weight Codewords



UNIVERSITY OF WATERLOO

Equality Operator for Constant-weight Codewords

Plain

x	x_{m-1}	x_{m-2}	...	x_2	x_1	x_0
y	0	1	...	0	0	1

$$x_{m-2} \times \cdots \times x_0 = e$$

Arithmetic

x	x_{m-1}	x_{m-2}	...	x_2	x_1	x_0
y	y_{m-1}	y_{m-2}	...	y_2	y_1	y_0

Equality Operator for Constant-weight Codewords

Plain

x	x_{m-1}	x_{m-2}	...	x_2	x_1	x_0
y	0	1	...	0	0	1

$$x_{m-2} \times \cdots \times x_0 = e$$

Arithmetic

x	x_{m-1}	x_{m-2}	...	x_2	x_1	x_0
y	y_{m-1}	y_{m-2}	...	y_2	y_1	y_0

Inner product

$$c \Rightarrow \frac{1}{k!}\prod_{i=0}^{k-1}(c - i) = e$$

UNIVERSITY OF WATERLOO

Equality Operator for Constant-weight Codewords

Plain

# of Mult = k
Mult. Depth = $\log_2 k$

x	x_{m-1}	x_{m-2}	...	x_2	x_1	x_0
y	0	1	...	0	0	1

$$x_{m-2} \times \cdots \times x_0 = e$$

Arithmetic

# of Mult = $m + k$
Mult. Depth = $\log_2 k + 1$

x	x_{m-1}	x_{m-2}	...	x_2	x_1	x_0
y	y_{m-1}	y_{m-2}	...	y_2	y_1	y_0

Inner product

$$c \Rightarrow \frac{1}{k!}\prod_{i=0}^{k-1}(c - i) = e$$

UNIVERSITY OF WATERLOO

Evaluation of Equality Operators

Constant-weight Plain Operator

- 10x faster than folklore
- Attributed to
  - Less operations
  - Smaller Parameters
- Larger representation size
  - Higher memory usage
  - Slower runtime for less memory

	n	2^8	2^{16}	2^{32}	2^{64}	2^{128}	2^{256}	2^{512}
Plain Folklore	ℓ	8	16	32	64	128	256	512
	Mult Depth	3	4	5	6	7	8	9
	$N = 8192$	0.27	0.54	-	-	-	-	-
	$N = 16384$	1.1	2.4	5.0	10	21	42	84
Plain Constant-weight $k = \frac{1}{4}\log_2 n$	k	2	4	8	16	32	64	128
	Mult Depth	1	2	3	4	5	6	7
	m	24	37	64	117	221	427	838
	$N = 4096$	0.01	-	-	-	-	-	-
	$N = 8192$	0.04	0.12	0.25	0.49	-	-	-
	$N = 16384$	0.17	0.48	1.1	2.4	5.0	10	21
Plain Constant-weight $k = \frac{1}{8}\log_2 n$	k	1	2	4	8	16	32	64
	Mult Depth	0	1	2	3	4	5	6
	m	256	363	569	968	1749	3290	6349
	$N = 4096$	**0.0001**	**0.008**	-	-	-	-	-
	$N = 8192$	0.0004	0.038	**0.10**	**0.25**	**0.54**	-	-
	$N = 16384$	0.002	0.17	0.5	1.1	2.4	**5.0**	**10**

UNIVERSITY OF WATERLOO

Evaluation of Equality Operators

Arithmetic Operators

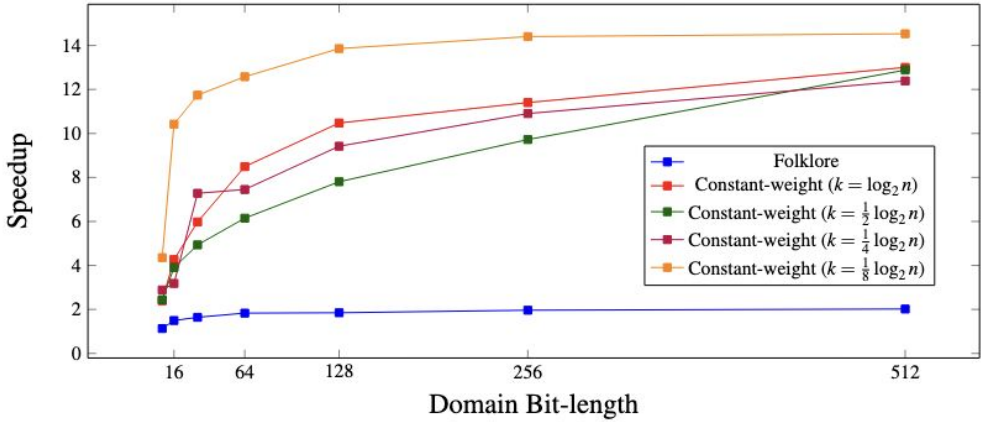
- Comparable runtime
- More operations for constant-weight
- Better when smaller parameters are used

	n	2^8	2^{16}	2^{32}	2^{64}	2^{128}	2^{256}	2^{512}
Arithmetic Folklore	ℓ	8	16	32	64	128	256	512
	Mult Depth	4	5	6	7	8	9	10
	$N = 8192$	**0.49**	-	-	-	-	-	-
	$N = 16384$	2.2	4.6	9.2	19	37	74	149
Arithmetic Constant-weight $k = \frac{1}{2}\log_2 n$	k	4	8	16	32	64	128	256
	Mult Depth	3	4	5	6	7	8	9
	m	11	19	36	68	132	261	517
	$N = 8192$	0.53	-	-	-	-	-	-
	$N = 16384$	2.2	4.3	**8.2**	**16**	**31**	**63**	**123**
Arithmetic Constant-weight $k = \frac{1}{4}\log_2 n$	k	2	4	8	16	32	64	128
	Mult Depth	2	3	4	5	6	7	8
	m	24	37	64	117	221	427	838
	$N = 8192$	0.85	**1.3**	-	-	-	-	-
	$N = 16384$	4.3	6.4	11	21	40	78	154

Evaluation of Equality Operators

Arithmetic Operators

- Comparable runtime
- More operations for constant-weight
- Better when smaller parameters are used
- Parallelization is better for constant-weight

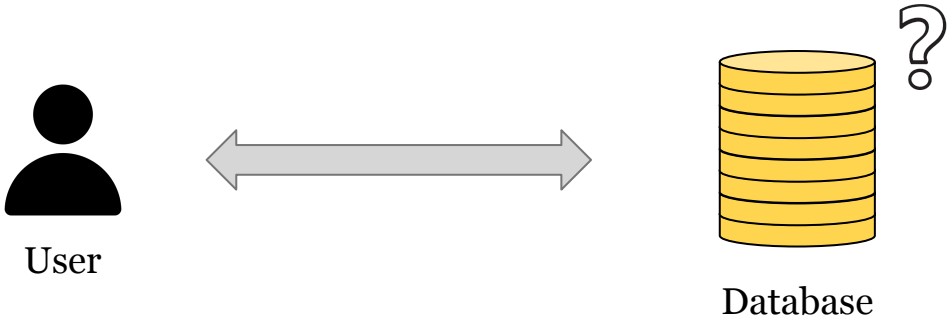


UNIVERSITY OF WATERLOO

Let’s Build a PIR Protocol

Private Information Retrieval

- Retrieve an element from a database without the server learning which element is retrieved



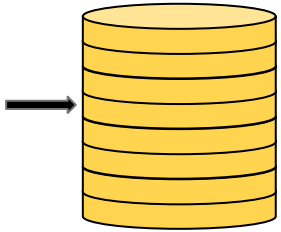
Private Information Retrieval

- Retrieve an element from a database without the server learning which element is retrieved
- Single server and multi-server
  - We focus on single-server PIR

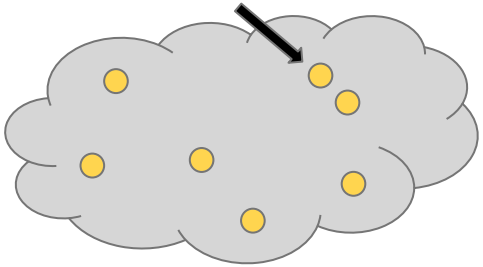
Private Information Retrieval

- Retrieve an element from a database without the server learning which element is retrieved
- Single server and multi-server
  - We focus on single-server PIR
- Variations:

Index PIR



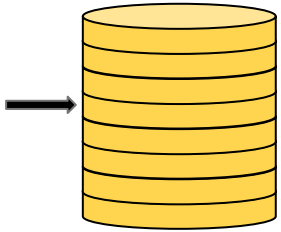
Keyword PIR



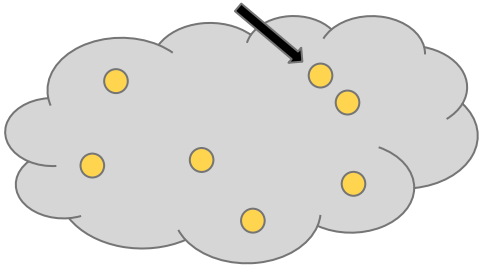
Private Information Retrieval

- Retrieve an element from a database without the server learning which element is retrieved
- Single server and multi-server
  - We focus on single-server PIR
- Variations:

Index PIR



Keyword PIR



Example
- **256-bit identifiers**
- **filenames**

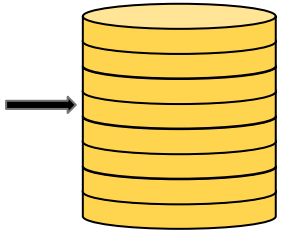
UNIVERSITY OF WATERLOO

Private Information Retrieval

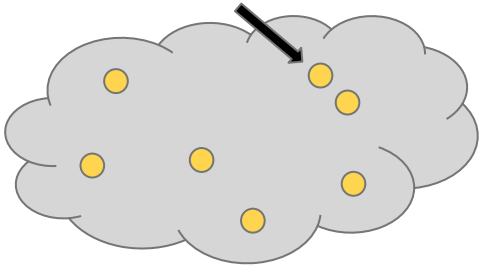
- Retrieve an element from a database without the server learning which element is retrieved
- Single server and multi-server
  - We focus on single-server PIR
- Variations:

SealPIR
MulPIR
.
.
.

Index PIR
(easier)



Keyword PIR
(harder)

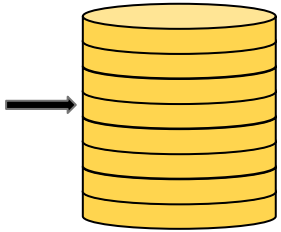


Private Information Retrieval

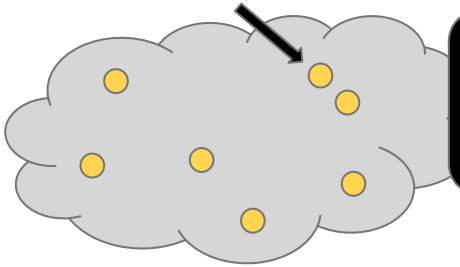
- Retrieve an element from a database without the server learning which element is retrieved
- Single server and multi-server
  - We focus on single-server PIR
- Variations:

SealPIR
MulPIR
.
.
.

Index PIR
(easier)



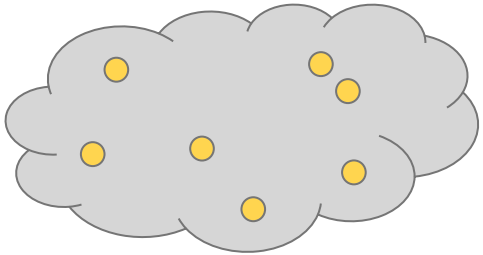
Keyword PIR
(harder)



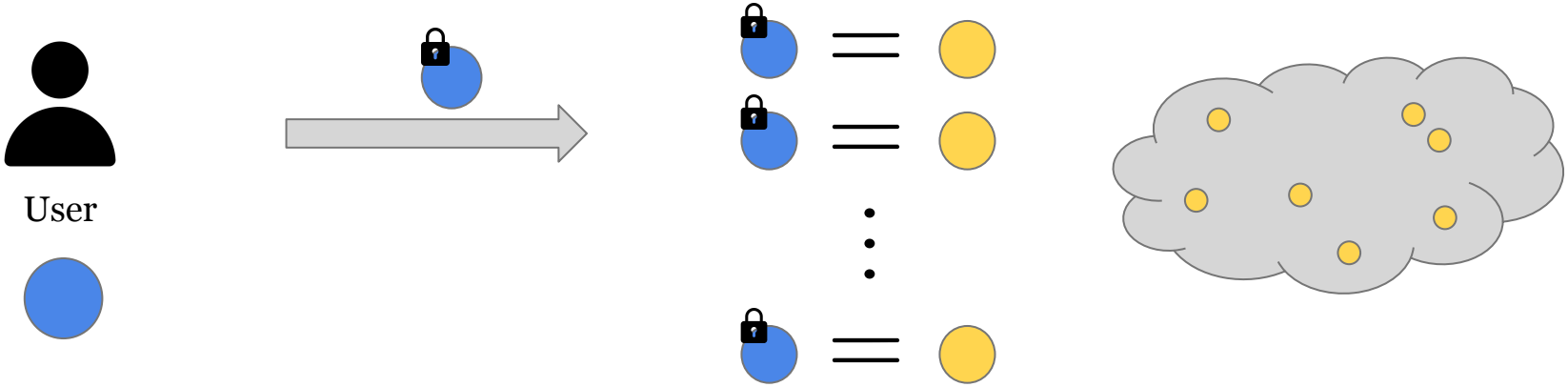
Extra round
Probabilistic Failure

UNIVERSITY OF WATERLOO

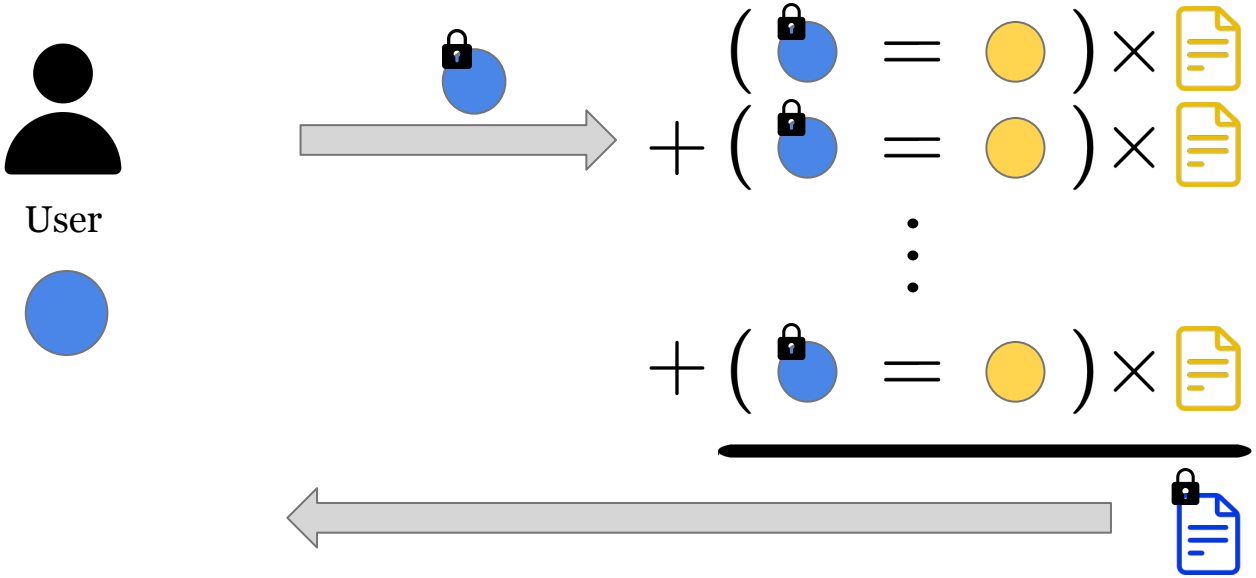
Constant-weight PIR



Constant-weight PIR

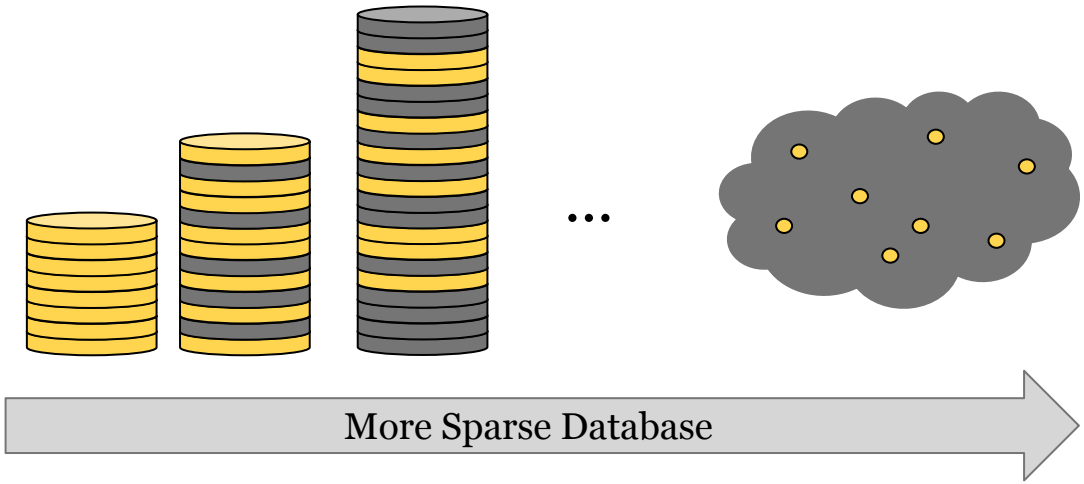


Constant-weight PIR



PIR for Sparse Databases

- Slower in a packed database
- Advantageous as domain grows



UNIVERSITY OF WATERLOO

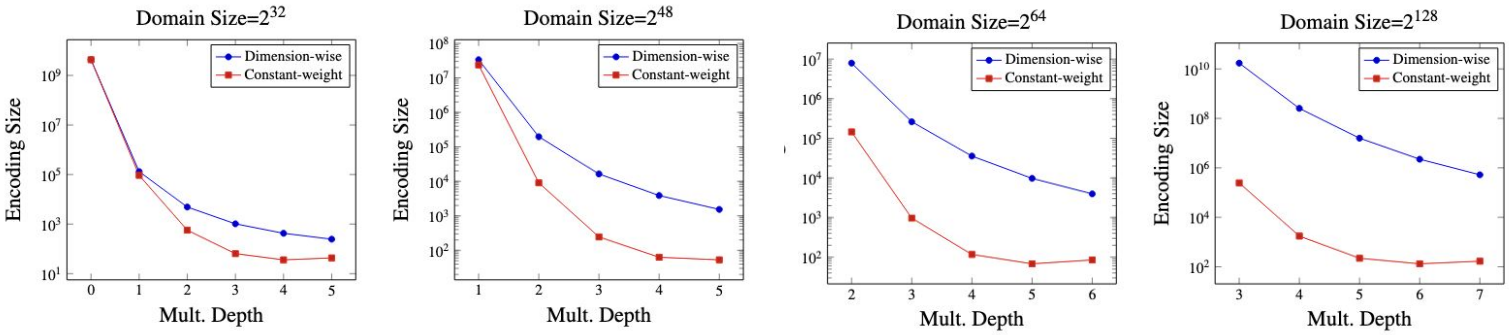
PIR for Sparse Databases

- Slower in a packed database
- Advantageous as domain grows
- # of operations doesn't depend on domain size

Method	Mult Depth	Query Bit-length	# of Operations (Excluding Expansion)	Download Cost (in cts)
SealPIR	$d-1$	$d\left\lceil \sqrt[d]{\lvert S\rvert} \right\rceil$	$n \cdot \mathtt{PM} + \sum_{i=1}^{d-1} \lvert S\rvert^{\frac{d-i}{d}} F^i \cdot \mathtt{PM}$	F^{d-1}
MulPIR	$d-1$	$d\left\lceil \sqrt[d]{\lvert S\rvert} \right\rceil$	$n \cdot \mathtt{PM} + \sum_{i=1}^{d-1} \lvert S\rvert^{\frac{d-i}{d}} \cdot \mathtt{M}$	1
CwPIR	$\lceil \log k \rceil$	$O\left(\sqrt[k]{k!\lvert S\rvert} + k\right)$	$nk \cdot \mathtt{M} + n \cdot \mathtt{PM}$	1

PIR for Sparse Databases

Smaller representation size for fixed multiplicative depth

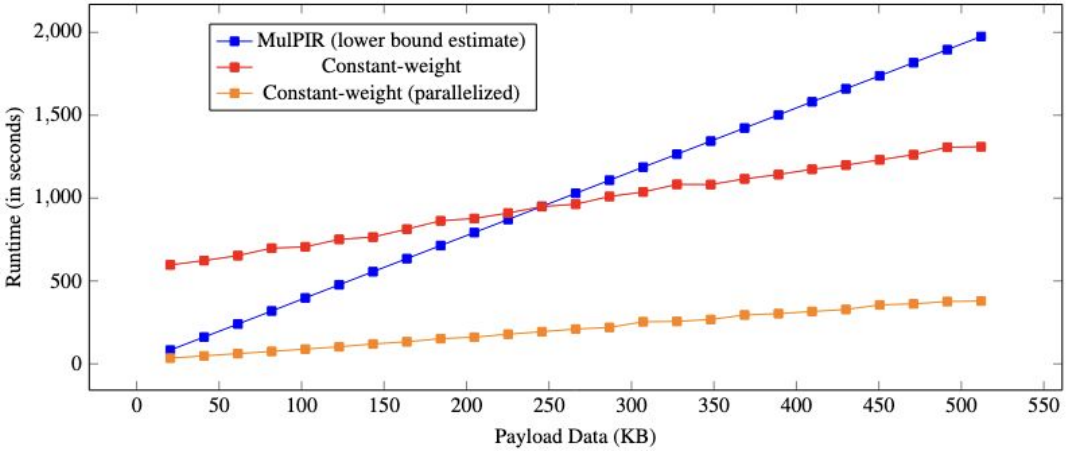


Encoding size as a function of multiplicative depth

UNIVERSITY OF WATERLOO

PIR with Large Response Size

- The response must fit into a ciphertext
- SealPIR/MulPIR are repeated when the response is large



UNIVERSITY OF WATERLOO

Takeaways

- Constant-weight Equality Operators
  - suitable for homomorphic encryption
  - faster than existing operators
- Constant-weight Keyword PIR
  - using equality operators
  - single-round
  - faster for sparse databases (i.e. keyword PIR)
  - faster for larger response sizes

Takeaways

- Constant-weight Equality Operators
  - suitable for homomorphic encryption
  - faster than existing operators
- Constant-weight Keyword PIR
  - using equality operators
  - single-round
  - faster for sparse databases (i.e. keyword PIR)
  - faster for larger response sizes

GitHub **RasoulAM/constant-weight-pir**

ARTIFACT EVALUATED usenix ASSOCIATION AVAILABLE

ARTIFACT EVALUATED usenix ASSOCIATION FUNCTIONAL

ARTIFACT EVALUATED usenix ASSOCIATION REPRODUCED

UNIVERSITY OF WATERLOO

UNIVERSITY OF WATERLOO

CrySP

Cryptography, Security, and Privacy Research Group

Thank you!

Questions?