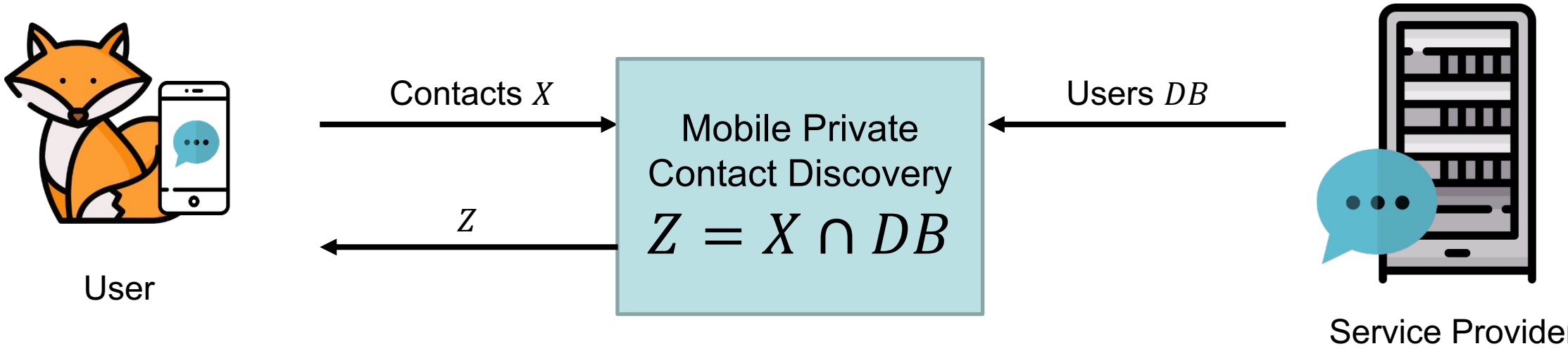
Scaling Mobile Private Contact Discovery to Billions of Users

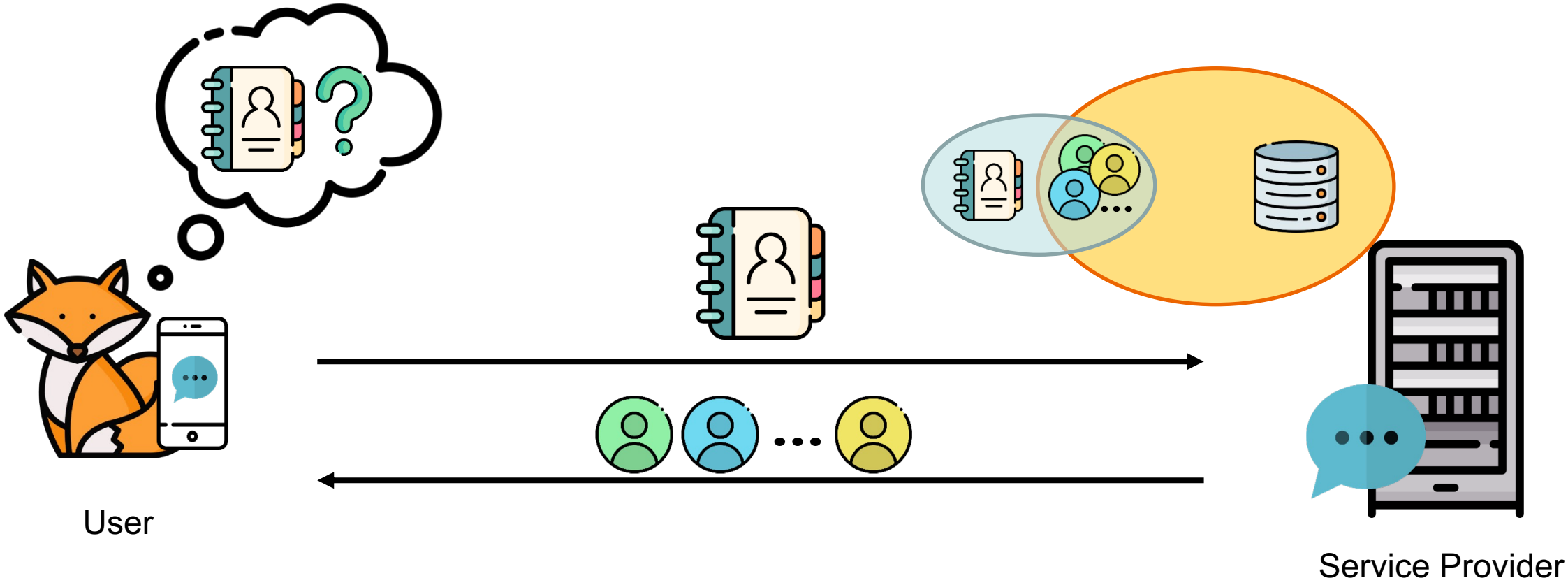
Laura Hetz, Thomas Schneider, Christian Weinert

TECHNISCHE UNIVERSITÄT DARMSTADT

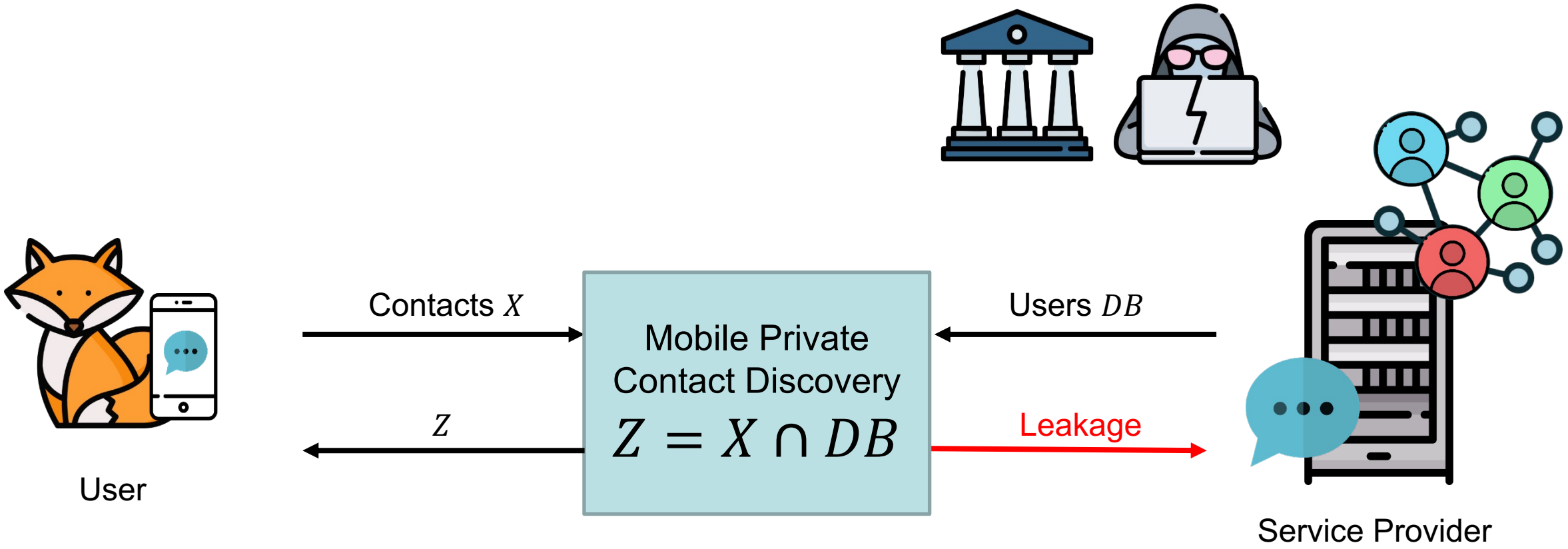


https://contact-discovery.github.io

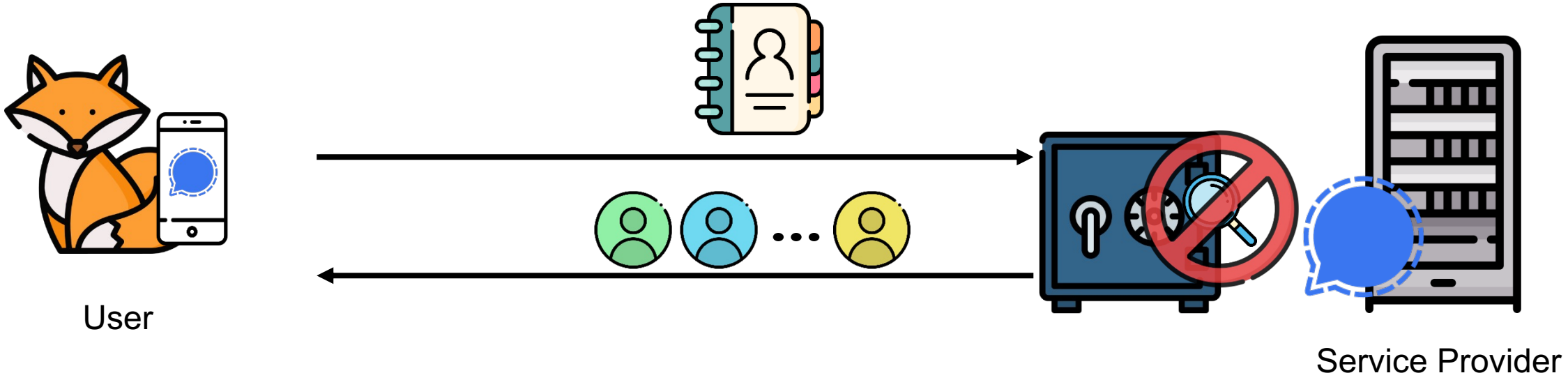
Mobile Contact Discovery – Motivation [KRS+19, HWS+21, HWS+23]

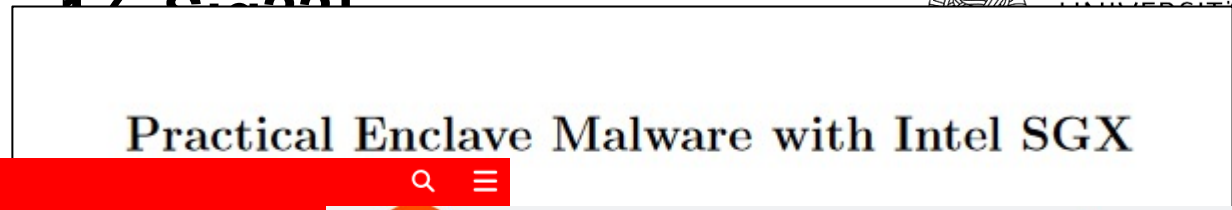


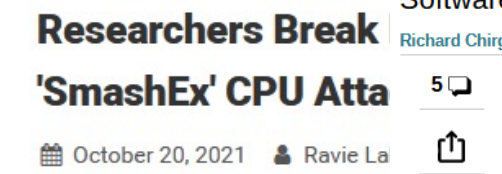
Mobile Contact Discovery – Motivation



Mobile Contact Discovery – Signal [Sig17, Sig22]







{* SECURITY *}

Boffins show Intel's SGX can leak crypto keys

Software Guard Extensions are supposed to hide data. But the 'Prime+Probe attack' fixe

Richard Chirgwin — Tue 7

5

A researcher who in January helped highlight possible flaws in Intel's Software Guard Extensions' input-output protection is back, this time with malware running inside a protected SGX enclave.

Instead of protecting the system, Samuel Weiser and four collaborators of

A newly disclosed vulnerability affecting Intel processors could be abused by an adversary to gain access to sensitive information stored within enclaves and even run arbitrary code on vulnerable systems.

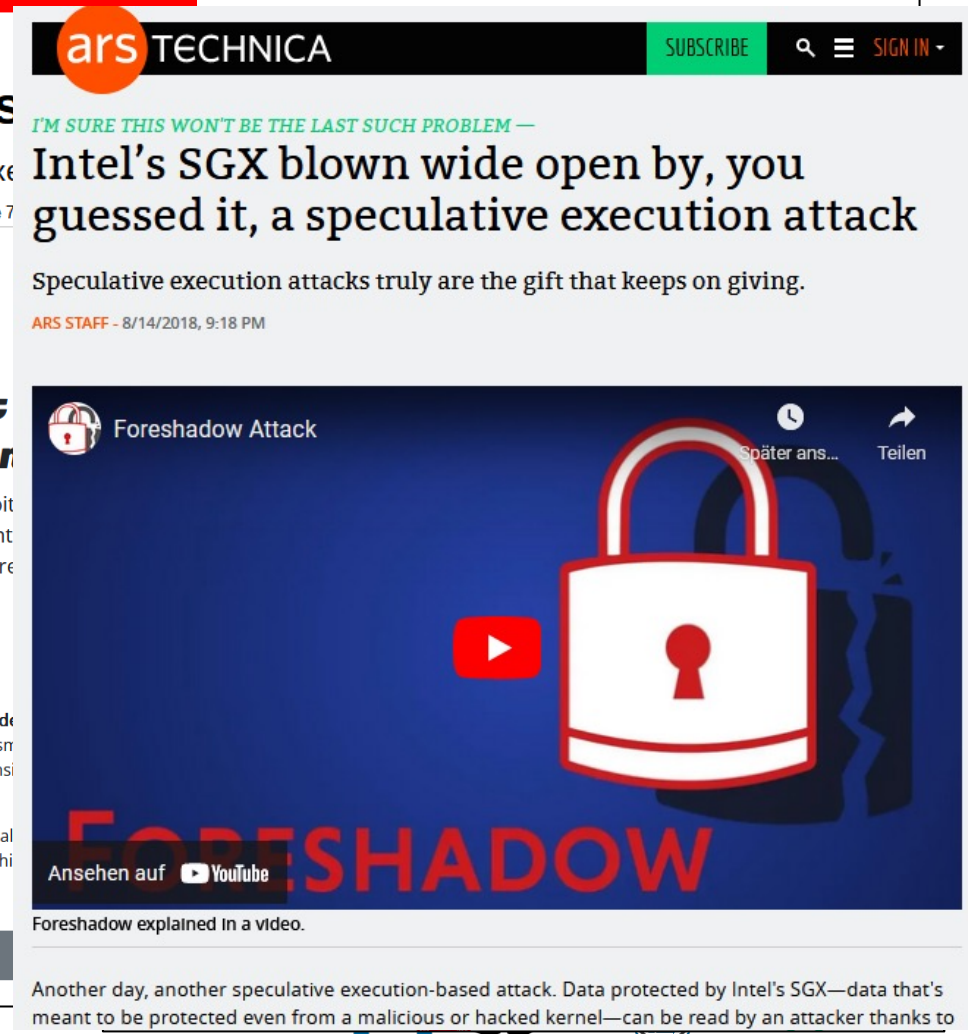
LVI - Hijacking Transient with Load Value Injection

LVI is a new class of transient-execution attacks exploit **flaws** in modern processors to **inject attacker data** int **steal sensitive data and keys** from Intel SGX, a secure processors for your personal data.

a extraction attacks around, like Meltdown, Foreshadow, ZombieLoad, RIDL and Fallout, and **de** directly leaking data from the victim to the attacker, we proceed in the opposite direction: we sm **hidden processor buffers** into a victim program and hijack transient execution to acquire sensi passwords.

arder to mitigate than previous attacks, as it can affect virtually any access to memory. Unlike al ransparently mitigated in existing processors and necessitates expensive **software patches**, whi ? up to 19 times.

Read | Cite



I'M SURE THIS WON'T BE THE LAST SUCH PROBLEM —

Intel's SGX blown wide open by, you guessed it, a speculative execution attack

Speculative execution attacks truly are the gift that keeps on giving.

ARS STAFF - 8/14/2018, 9:18 PM

Foreshadow explained in a video.

Another day, another speculative execution-based attack. Data protected by Intel's SGX—data that's meant to be protected even from a malicious or hacked kernel—can be read by an attacker thanks to

PSI FOR PRIVATE CONTACT DISCOVERY

High-Level Idea: OPRF-based PSI

Input: Contacts X

Store encrypted database

DB'

Generate secret key k

Encrypt DB records with key k:

$DB[j]' = PRF_k(DB[j])$

Input: DB

Run OPRF for all contacts x_i

Check if e_i is in DB'

Related Work – Optimization of OPRF-based PSI [KLS+17, RA18, KRS+19]

Input: Contacts X

1. Base Phase

OPRF Precomputation

Generate secret key k

Precomputation

Input: DB

2. Setup Phase

Store Cuckoo filter CF

CF

Encrypt DB records with key k

Insert items into Cuckoo filter CF

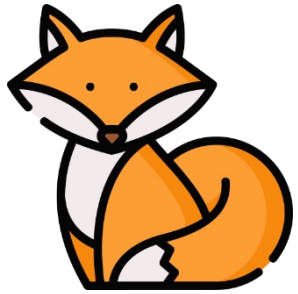
3. Online Phase

Run OPRF for all contacts x_i

Check if e_i is in CF

~ 8 GiB communication for 2^{31} registered users

Related Work – Combining PIR & PSI [DRRT18]



Input: Contacts X

$query$

idx

$DB[idx]$
$+leakage(DB)$

PIR

DB

+

PSI

$answer$

Reconstruct intersection from answer

$query$

$answer$

Input: DB

PIR-based CF Lookup in PSI [DRRT18, KRS+19]

Input: Contacts X

1. Base Phase

OPRF Precomputation

2. Setup Phase

PIR Preprocessing

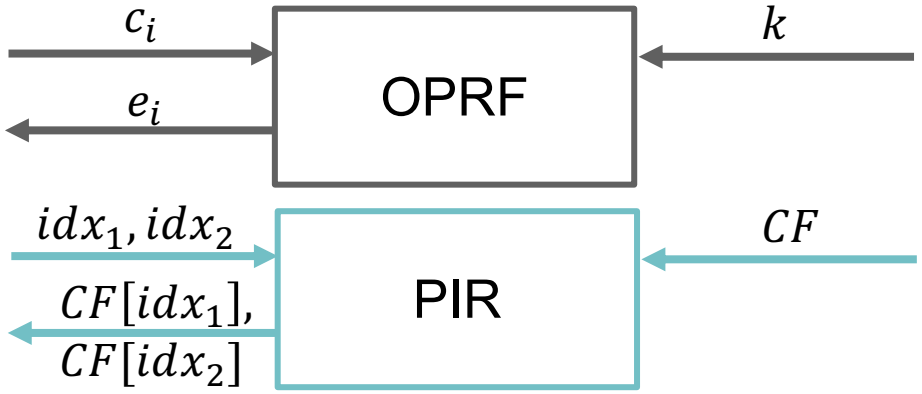
PIR Pre-processing

Input: DB

3. Online Phase

Compute possible CF positions idx_1, idx_2 for element e_i

Check if element e_i is in received buckets



PIR-based CF Lookup in PSI [DRRT18, KRS+19]

Input: Contacts X

1. Base Phase

OPRF Precomputation

2. Setup Phase

PIR Preprocessing

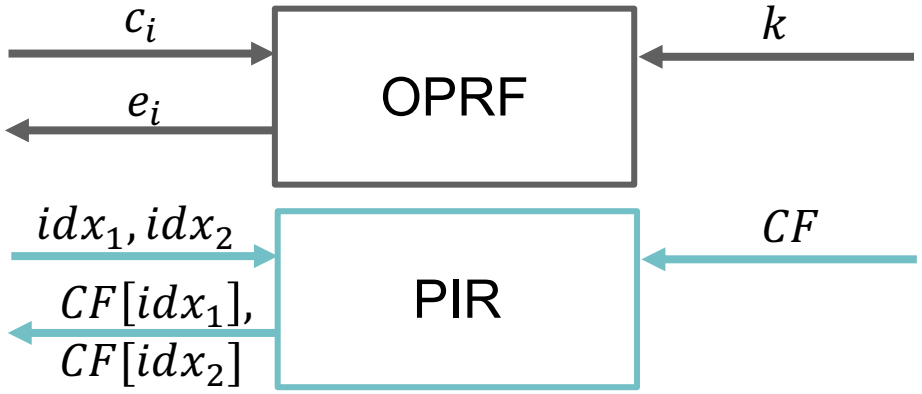
PIR Pre-processing

Input: DB

3. Online Phase

Compute possible CF positions idx_1, idx_2 for element e_i

Check if element e_i is in received buckets



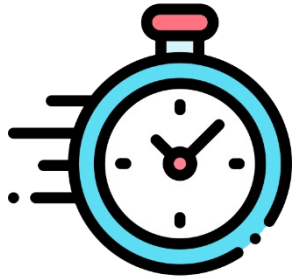
Requirements – PIR Selection – Handling Large Set Sizes – Query Scheduling

SYSTEM DESIGN

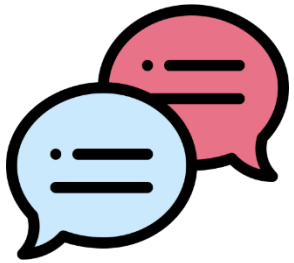
Requirements

Database Size up to 2^{31}

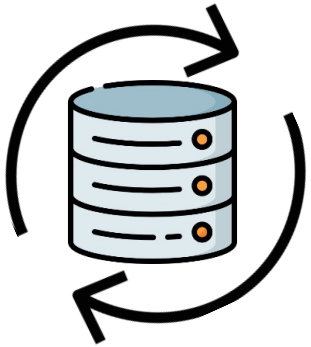
Up to 2^{10} **client contacts**



Fast online phase



Low Communication

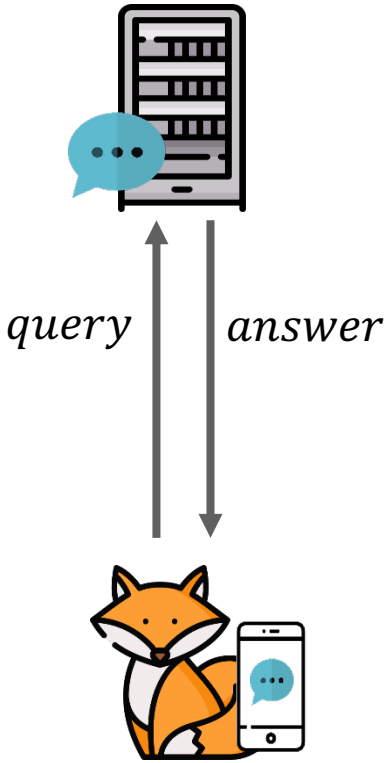


DB Updates

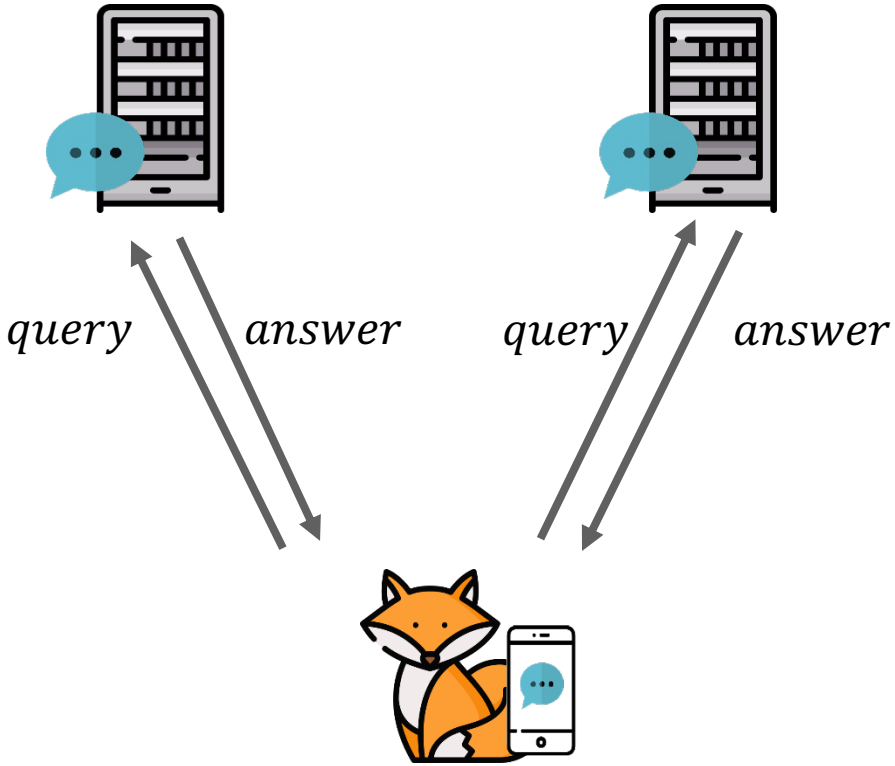
Requirements – PIR Selection – Handling Large Set Sizes – Query Scheduling

SYSTEM DESIGN

PIR Protocol Selection

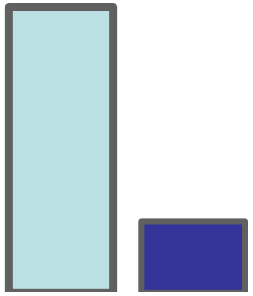


Single-Server PIR
[ACLS18, CK20, ALP+21, MCR21, CHK22, DPC23, HHC+23, LLWR24, LP22, MW22, ZLTS23, MR23]

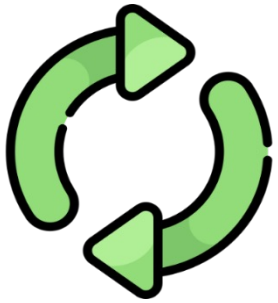


Multi-Server PIR
[BGI15, BGI16, CK20, KC21, SACM21, GHPS22, MZRA22]

Offline-Online PIR – Advantages [CK20, KC21, MZRA22]



Online comm. $O(\log |DB|)$ and comp. $O(\sqrt{|DB|})$



Amortization of offline costs over many queries

Database updates without rerunning offline phase

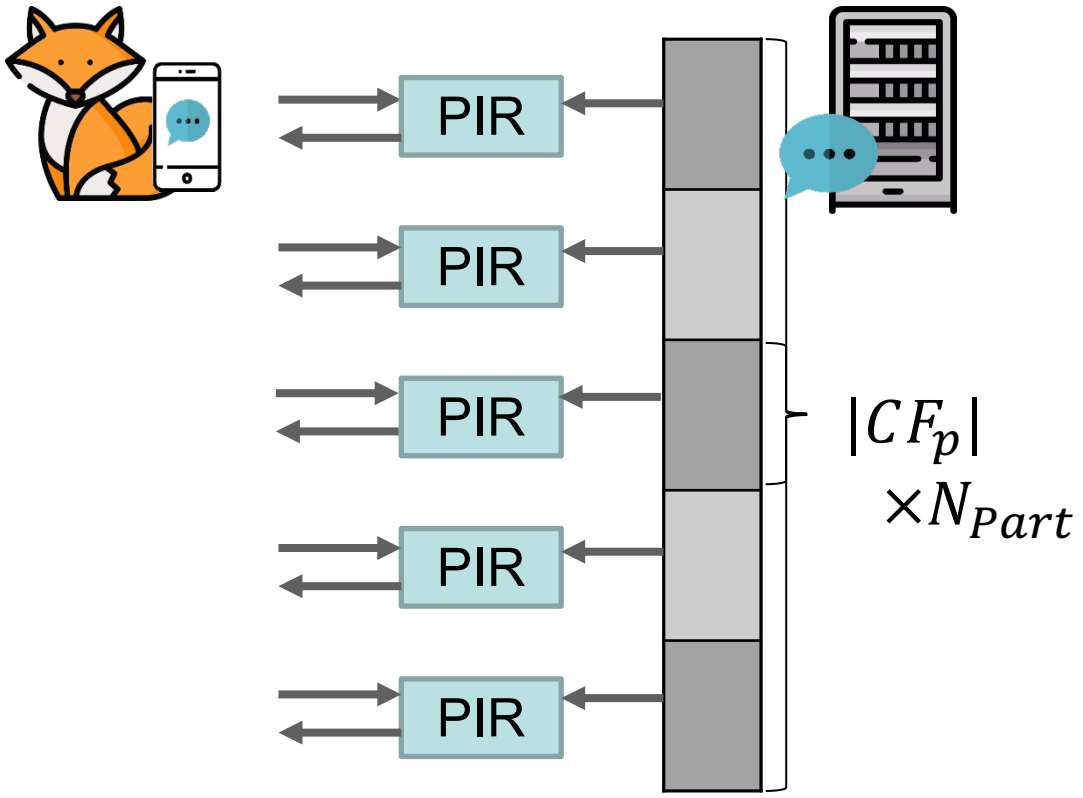
Batching possible

Implementation for mobile devices

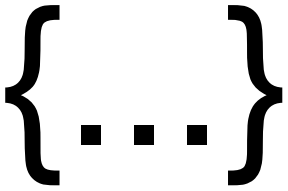
Requirements – PIR Selection – Handling Large Set Sizes – Query Scheduling

SYSTEM DESIGN

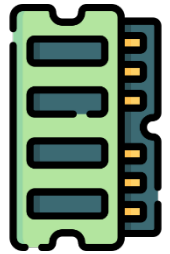
Handling Large Set Sizes



Offline time
$O(|DB|)$



Set size
$\sqrt{|DB|}$



Memory requirements

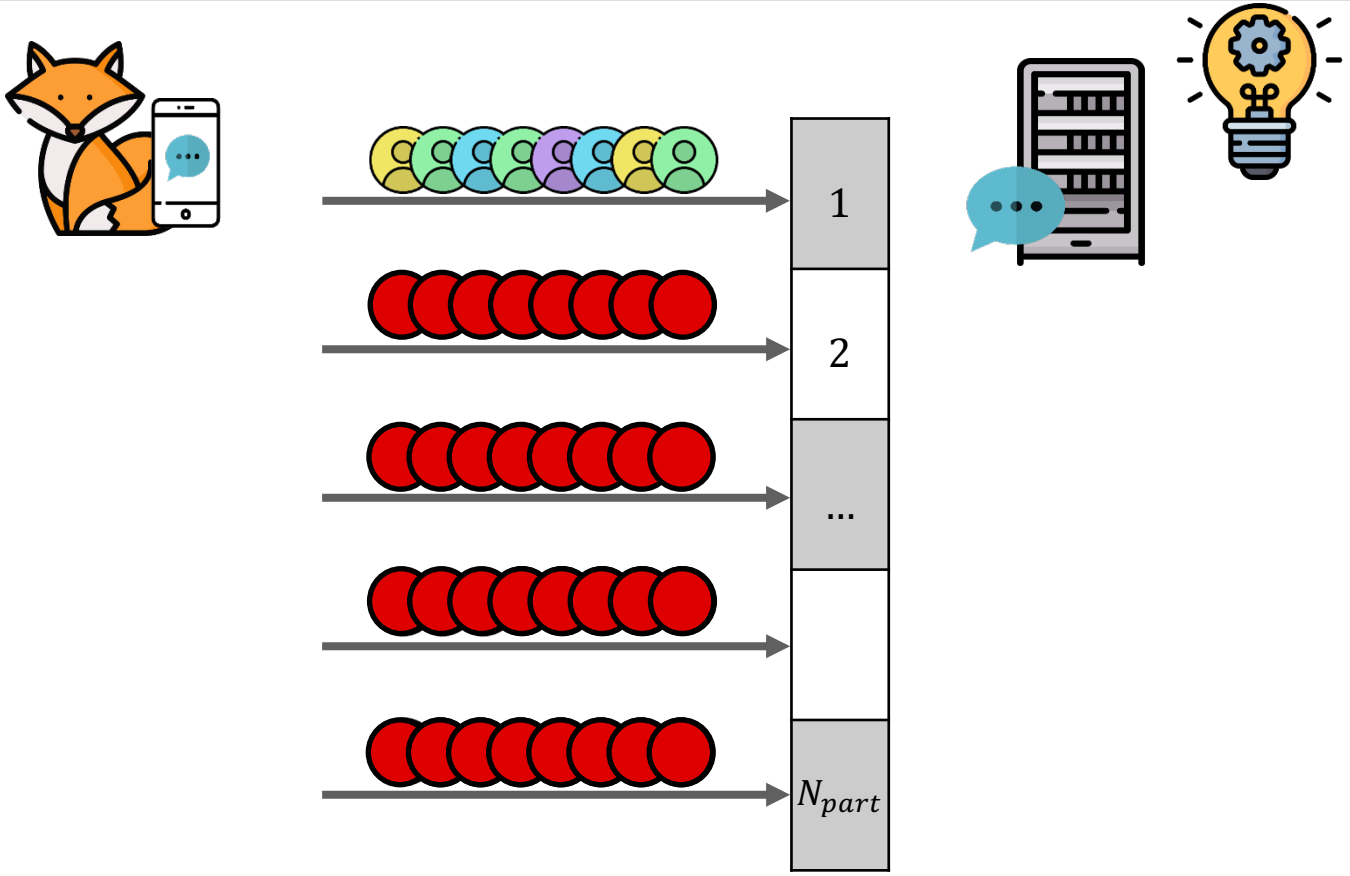
Not supported in implementation

Inefficient batching

Requirements – PIR Selection – Handling Large Set Sizes – Query Scheduling

SYSTEM DESIGN

Query Scheduling – Naive Approach



Worst Case:

For $|DB| = 2^{31}$,
$|X| = 2^{10}$,
$N_{Part} = 64$:

⇒ 73 MiB communication

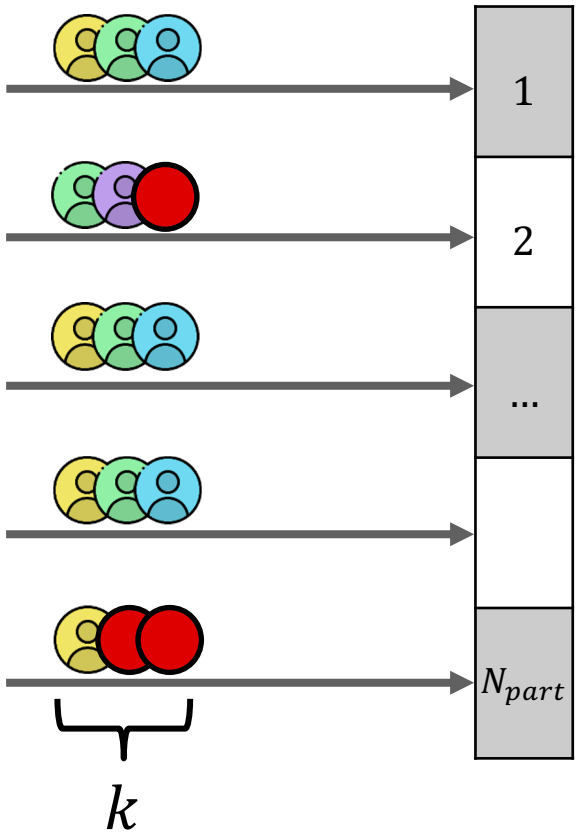
Query Scheduling – Simple Hashing [PSSZ15, PSZ18, DRRT18]

$$p = 1 - \left(\sum_{i=0}^{k-1} \binom{2|X|}{i} \cdot \left(\frac{1}{N_{Part}}\right)^{i} \cdot \left(1 - \frac{1}{N_{Part}}\right)^{2|X|-i} \right)^{N_{Part}}$$

For $|DB| = 2^{31}$,
$|X| = 2^{10}$,
$N_{Part} = 64$:

⇒ 3 MiB communication

➔ 24× less communication than naïve approach

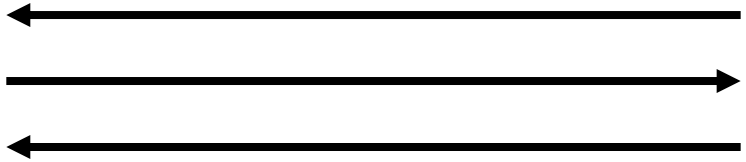


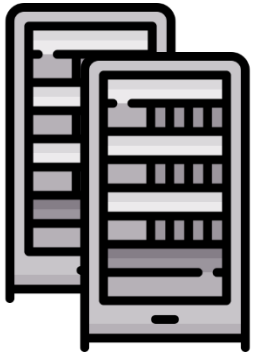
IMPLEMENTATION & EVALUATION

See https://encrypto.de/code/disco

Experiments

$|X| \leq 2^{10}$





$|DB| \leq 2^{31}$

OnePlus 8T
Snapdragon 865,
12 GB RAM,
256 GB Storage

Two Servers
128 GB RAM,
8 virtual cores,
250 GB Storage

8× Larger set size than related work

PSI Comparison – [KRS+19] & Our PSI Protocol

TECHNISCHE UNIVERSITÄT DARMSTADT

Parameters				Base		Setup			Online	
$\|DB\|$	$\|X\|$	OPRF	PSI	Time [s]	Comm. [MiB]	Time S [h]	Time C [s]	Comm. [MiB]	Time [s]	Comm. [MiB]
2^{28}	2^{10}	NR-ECC	[KRS+19]	0.15	2.04	9.84	3.53	1 072.14	2.20	4.05
			Ours	0.15	2.04	9.90	109.63	66.00	5.59	5.95
		GC-LowMC	[KRS+19]	1.26	21.56	0.55	3.53	1 072.14	0.63	2.02
			Ours	1.26	21.56	0.61	109.63	66.00	4.02	3.92

less setup communication for $|DB| = 2^{31}$

S: Server
C: Client

CS ENCRYPTO CRYPTOGRAPHY AND PRIVACY ENGINEERING

PSI Comparison – PIR-PSI [DRRT18] & Our PSI Protocol

TECHNISCHE UNIVERSITÄT DARMSTADT

Parameters				Offline			Online		
$\|DB\|$	$\|X\|$	Protocol PSI	Params.	Time [s] ST	Time [s] MT	Comm. [MiB]	Time [s] ST	Time [s] MT	Comm. [kiB]
2^{28}	2^{10}	PIR-PSI	$c = 0.25, b = 1$	–	–	–	33.02	13.22	5 048.32
			$c = 4, b = 16$	–	–	–	4.07	1.60	28 979.20
		Ours	$N_{Part} = 32$	326.10	98.97	66.00	3.39	0.45	1 952.25

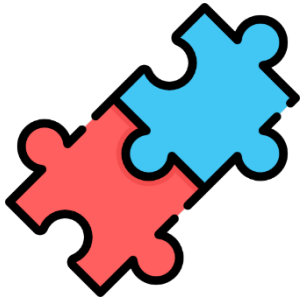
< 2 seconds online time for MT, $|DB| = 2^{31}, |X| = 2^{10}$

ST: single-threaded
MT: multi-threaded

CS ENCRYPTO CRYPTOGRAPHY AND PRIVACY ENGINEERING

Summary

Surveyed PIR protocols



Optimized & integrated PIR into OPRF-based PSI

Simulated & **evaluated strategies** for DB updates

Implemented our protocol for mobile devices

Ran experiments and **evaluated** our protocol

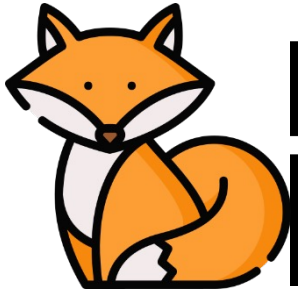
Limitations & Outlook

Thank you!

TECHNISCHE UNIVERSITÄT DARMSTADT

Get in touch:
laura.hetz@inf.ethz.ch

For more details see https://ia.cr/2023/758
& https://contact-discovery.github.io



CS

ENCRYPTO CRYPTOGRAPHY AND PRIVACY ENGINEERING

References

[ACLS18] S. Angel, H. Chen, K. Laine, and S. Setty. “PIR with Compressed Queries and Amortized Query Processing”. In: S&P 2018.

[ALP+21] A. Ali, T. Lepoint, S. Patel, M. Raykova, P. Schoppmann, K. Seth, K. Yeo. “Communication-Computation Trade-offs in PIR”. In: USENIX Security 2021.

[BGI15] E. Boyle, N. Gilboa, and Y. Ishai. “Function Secret Sharing”. In: EUROCRYPT 2015.

[BGI16] E. Boyle, N. Gilboa, and Y. Ishai. “Function Secret Sharing: Improvements and Extensions”. In: ACM CCS 2016.

[CHK22] H. Corrigan-Gibbs, A. Henzinger, and D. Kogan. “Single-Server Private Information Retrieval with Sublinear Amortized Time”. In: EUROCRYPT 2022.

[CK20] H. Corrigan-Gibbs and D. Kogan. “Private Information Retrieval with Sublinear Online Time”. In: EUROCRYPT 2020.

[DRRT18] D. Demmler, P. Rindal, M. Rosulek, and N. Trieu. “PIR-PSI: Scaling Private Contact Discovery”. In: PETS 2018.

[DPC23] A. Davidson, G. Pestana, S. Celi. “FrodoPIR: Simple, Scalable, Single-Server Private Information Retrieval”. In: PETS 2023.

[FAKM14] B. Fan, D. Andersen, M. Kaminsky, and M. D. Mitzenmacher. "Cuckoo Filter: Practically Better Than Bloom“. In: ACM CoNEXT 2014.

[GHPS22] D. Günther, M. Heymann, B. Pinkas, and T. Schneider. “GPU-accelerated PIR with Client-Independent Preprocessing for Large Scale Applications”. In: USENIX Security 2022.

References

[HHC+23] A. Henzinger, M. M. Hong, H. Corrigan-Gibbs, S. Meiklejohn, and V. Vaikuntanathan. “One Server for the Price of Two: Simple and Fast Single-Server Private Information Retrieval“. In: USENIX Security 2023.

[HWS+21] C. Hagen, C. Weinert, C. Sendner, A. Dmitrienko, T. Schneider. “All the Numbers are US: Large-scale Abuse of Contact Discovery in Mobile Messengers”. In: NDSS 2021.

[HWS+23] C. Hagen, C. Weinert, C. Sendner, A. Dmitrienko, T. Schneider. “Contact Discovery in Mobile Messengers: Low-cost Attacks, Quantitative Analyses, and Efficient Mitigations”. In: ACM TOPS 2023.

[KC21] D. Kogan and H. Corrigan-Gibbs. “Private Blocklist Lookups with Checklist”. In: USENIX Security 2021.

[KLS+17] Á. Kiss, J. Liu, T. Schneider, N. Asokan, B. Pinkas. “Private Set Intersection for Unequal Set Sizes with Mobile Applications”. In: PETS 2017.

[KRS+19] D. Kales, C. Rechberger, T. Schneider, M Senker and C. Weinert. ”Mobile Private Contact Discovery at Scale”. In: USENIX Security 2019.

[LLWR24] J. Liu, J. Li, D. Wu, K. Ren. “PIRANA: Faster (Multi-query) PIR via Constant-weight Codes”. S&P 2024.

[LP22] A. Lazzaretti and C. Papamanthou. “Single Server PIR with Sublinear Amortized Time and Polylogarithmic Bandwidth”. Cryptology ePrint Archive. 2022.

[MCR21] M. H. Mughees, H. Chen, L. Ren. “OnionPIR: Response Efficient Single-Server PIR”. In: ACM CCS 2021.

References

[MR23] M. H. Mughees, L. Ren. “Vectorized Batch Private Information Retrieval”. In: S&P 2023.

[MW22] S. J. Menon, D. J. Wu. “SPIRAL: Fast, High-Rate Single-Server PIR via FHE Composition”. In: S&P 2022.

[MZRA22] Y. Ma, K. Zhong, T. Rabin, S. Angel. “Incremental Offline/Online PIR”. In: USENIX Security 2022.

[PSSZ15] B. Pinkas, T. Schneider, G. Segev, M. Zohner. “Phasing: Private Set Intersection Using Permutation-based Hashing”. In: USENIX Security 2015.

[PSZ18] B. Pinkas, T. Schneider, M. Zohner. “Scalable Private Set Intersection Based on OT Extension”. In: ACM TOPS 2018.

[RA18] A. C. D. Resende, D. F. Aranha, “Faster Unbalanced Private Set Intersection”. FC 2018.

[SACM21] E. Shi, W. Aqeel, B. Chandrasekaran, and B. Maggs. “Puncturable Pseudorandom Sets and Private Information Retrieval with Polylogarithmic Bandwidth and Sublinear Time”. CRYPTO 2021.

[Sig17] M. Marlinspike. “Technology preview: Private contact discovery for Signal”. https://signal.org/blog/private-contact-discovery/. 2017.

[Sig22] G. Connell. “Technology Deep Dive: Building a Faster ORAM Layer for Enclaves”. https://www.signal.org/blog/building-faster-oram/. 2022.

[ZLTS23] M. Zhou, W.-K. Lin, Y. Tselekounis, E. Shi. “Optimal Single-Server Private Information Retrieval”. EUROCRYPT 2023.

This presentation has been designed using resources from Flaticon.com, Android.com, Signal.org, Telegram.org, Whatsapp.com, and Oneplus.com.

BACKUP SLIDES

PIR Protocol Selection

TECHNISCHE UNIVERSITÄT DARMSTADT

# of servers (n)	Protocol	Sec. Assumption	Preprocessing	Updatablity	Batching	Implementation	Offline Comp. C	Offline Comp. S	Offline Comm.	Online Comp. C	Online Comp. S	Online Comm.
1	SealPIR [ACLS18]	RLWE	✓	✗	✓	✓	-	N	-	N		$dN^{1/d}$
	MulPIR [ALP+21]	RLWE	✓	✗	✓	✓•	-	N	-			$dN^{1/d}$
	[MR23]	RLWE	✓‡	✗	✓	✓	-	N	-	$B/pN_B^{2/d}$		$BN_B^{1/d}/p$
	Spiral (family) [MW22]	RLWE	✓	✗	✓	✓	-	N				$\log N$
	PIRANA [LLWR22]	RLWE	✓	✗	✓	✓•	-	N		N/M	N/M	N/M
	[CK20]	LWE	✓†	✗	✗	✗	$\sqrt{N}$	N	$\sqrt{N}$	$\sqrt{N}$		$\sqrt{N}$
	OnionPIR [MCR21]	RLWE	✓†	✗	✓	✓	N	N	N	N		N
	[LP22]	LWE	✓†	✗	✗	✗	N	N	N	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$
	[ZLTS22]	LWE	✓†	✗	✓	✗		N	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$	1
	[CHK22]	RLWE	✓†	✗	✓	✗	N	N	N	N	N	N
	SimplePIR [HHC+22]	LWE	✓†‡	✓§	✓	✓	N/M	N	$\sqrt{N}$		N	$\sqrt{N}$
	DoublePIR [HHC+22]	LWE	✓†‡	✓§	✓	✓		N	d_l^2			$\sqrt{N}$
	FrodoPIR [DPC22]	LWE	✓†‡	✓	✓	✓	N	N	1	N	1	N
2+	DPF-PIR [BGI15]	OWF	✗	-	✓	✓	-	-	-	$\log N$	N	$\log N$
	CIP-PIR [GHPS22]	OWF	✓‡	✓‡	✓	✓	-	N	-	N/n	$\sqrt{N}/n$	$n\sqrt{N}/n$
	[CK20]	OWF	✓†	✗	✗	✗	$\sqrt{N}$	N	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$	$\log N$
	[KC21]	OWF	✓†	✓¶	✓	✓⋆	$\sqrt{N}$	N	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$	$\log N$
	[SACM21]	LWE	✓†	✗	✓	✗	$\sqrt{N}$	N	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$	$\log N$
	iCK [MZRA22]	OWF	✓†	✓‖	✓	✓	$\sqrt{N}$	N	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$
	iSACM [MZRA22]	LWE	✓†	✓‖	✓	✓	$\sqrt{N}$	N	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$

Database size N,
number of servers n,
plaintext size p,
lattice dimension d_l,
database hypercube dimension d,
encryption parameter M,
number of buckets B, and
bucket size N_B.

† Stateful / offline-online
‡ Client-independent
§ Requires recomputation of hints for changed DB rows
¶ Waterfall updates
‖ In-place edits
⋆ Includes mobile implementation
• Implementation not public

CS ENCRYPTO CRYPTOGRAPHY AND PRIVACY ENGINEERING

Offline-Online PIR – Offline Phase [CK20, KC21]

$\boldsymbol{Server_{offline}}$

Input: DB,
Output: $\perp$

$\boldsymbol{SETUP}(DB)$

- Sample random subsets $S_1, \ldots, S_T \subset |DB|$ of size $|S_i| = \sqrt{|DB|}$
- Compute parity words p_j for each subset, store in $hints \leftarrow (S_1, p_1), \ldots (S_T, p_T)$

$\boldsymbol{Server_{online}}$

Input: DB,
Output: $\perp$

$hints$

Store $hints$ locally

Input: idx,
Output:$DB[idx]$

Offline-Online PIR – Online Phase [CK20, KC21]

$\boldsymbol{Server_{offline}}$

$\boldsymbol{ANSWER}(DB, query)$
Calculate parity for received punctured set $query$ and send it to client

$\boldsymbol{Server_{online}}$

$query_{offline}$

$answer_{offline}$

$query_{online}$

$answer_{online}$

$\boldsymbol{QUERY}(hints, idx)$

- Find subset S_i, s. t. $idx \in S_i$
- $query_{online} \leftarrow Punc(set_i, idx)$
- Generate new random set set_{new} with $idx \in set_{new}$
- $query_{offline} \leftarrow Punc(set_{new}, idx)$

$\boldsymbol{RECONSTRUCT}(answer_{offline}, answer_{online})$

- $p_{Set_i} \oplus answer_{online} = \begin{cases} DB[\text{idx}] \text{ if } idx \in DB \\ 0^{\ell} \text{ else} \end{cases}$
- Calculate new parity for set_{new} from $answer_{offline}$, save it, discard set_i

Protocol Overview – Offline Phase

$\boldsymbol{Server_{offline}}$

Input: DB_{PSI},
Output: ⊥

$\boldsymbol{SETUP}(DB)$

- Generate key k
- OPRF Precomputation
- Store DB in CF using PRF and key k
- Partition CF
- Generate $hints$ on N_{Part} partitions

CF

$\boldsymbol{Server_{online}}$

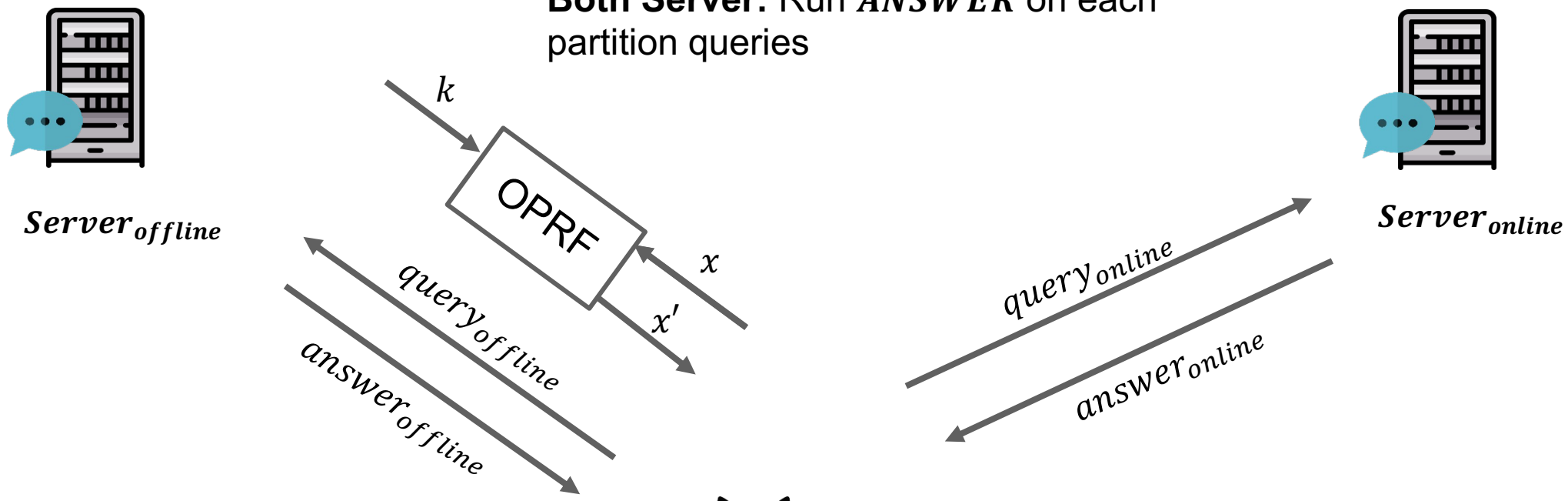
Input: ⊥,
Output: ⊥

$hints$

Store $hints$ locally

Input: $x \in X$,
Output:

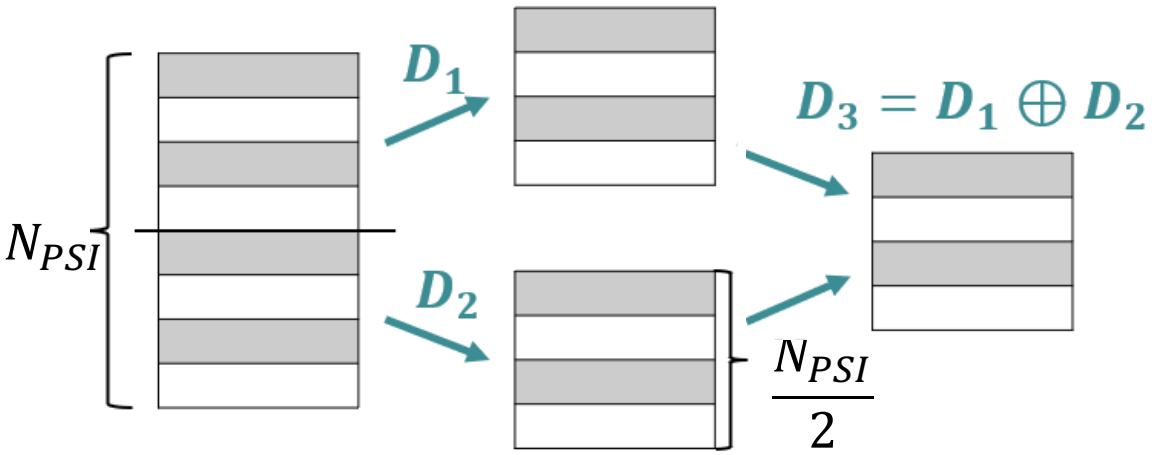
Protocol Overview – Online Phase



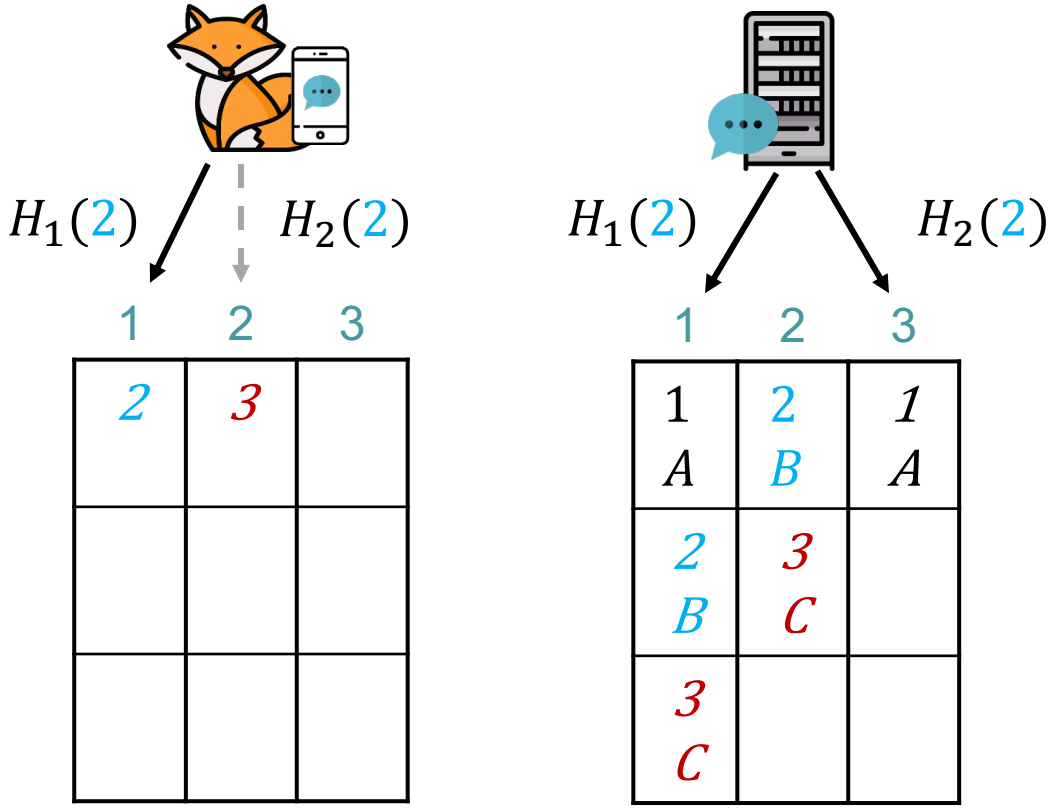
- Run OPRF with $Server_{offline}$ to get $x' = PRF_k(x)$
- Compute tag and all possible CF positions of x'
- Run query scheduling
- ***Query*** every partition

- Run ***RECONSTRUCT*** for all queries

Query Scheduling



Simple Construction [IKOS04]



Probabilistic Batch Codes [ACLS18]

PIR Selection – Handling Large Set Sizes – Query Scheduling – Protocol Overview

SYSTEM DESIGN

Protocol Overview – Offline Phase

$\boldsymbol{Server_{offline}}$

Input: DB_{PSI},
Output: $\perp$

$\boldsymbol{SETUP}(DB)$

- Generate key k
- OPRF Precomputation
- Store DB in CF using PRF and key k
- Partition CF
- PIR precomputation on each partition CF_p for $p \in \{1, N_{Parts}\}$

CF

$\boldsymbol{Server_{online}}$

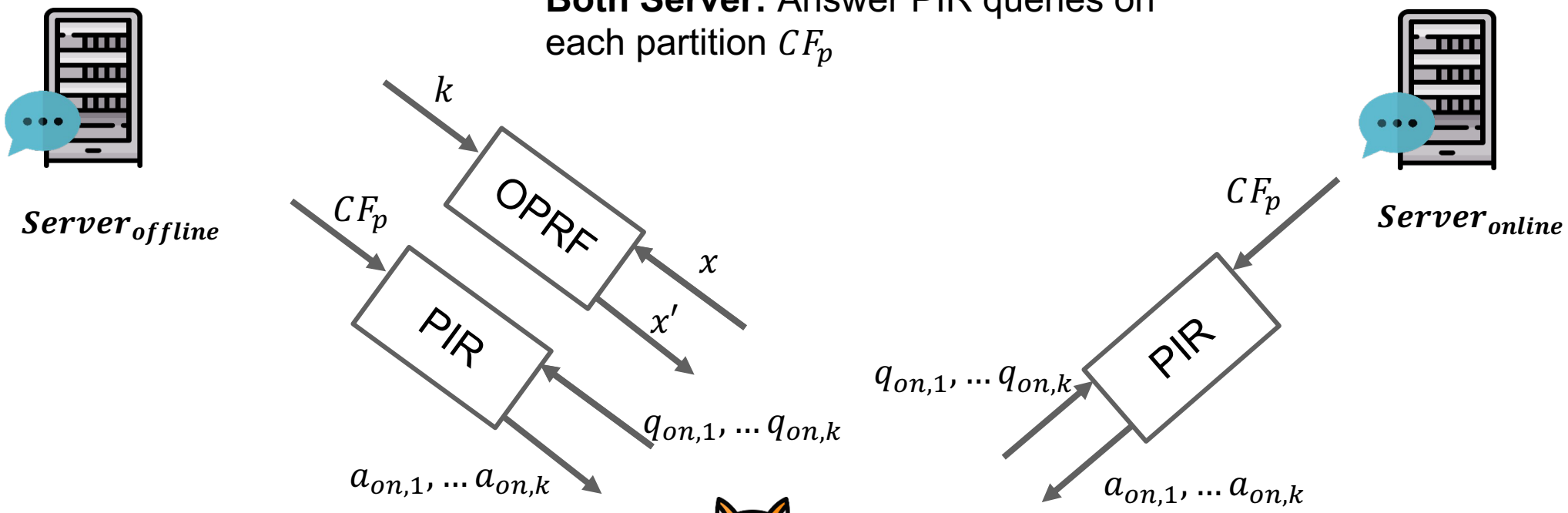
Input: $\perp$,
Output: $\perp$

$hints$

Store $hints$ locally

Input: $x \in X$,
Output:

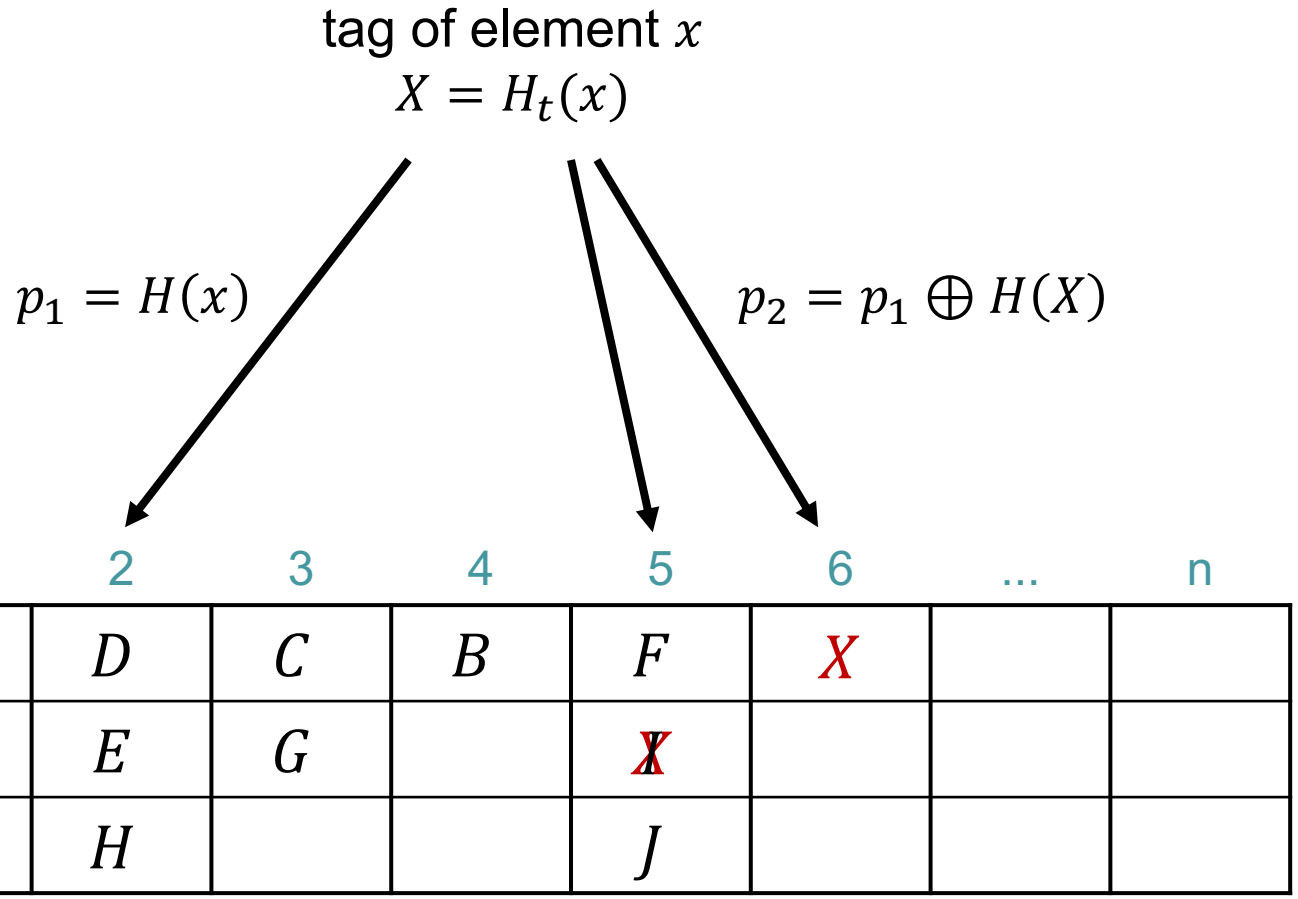
Protocol Overview – Online Phase



- Run **OPRF** to get $x' = PRF_k(x)$
- Compute tag and possible ***CF* positions** of x'
- Schedule queries to obtain k queries to each partitions $CF_p \in \{CF_1, \dots CF_{N_{Part}}\}$ for both server.

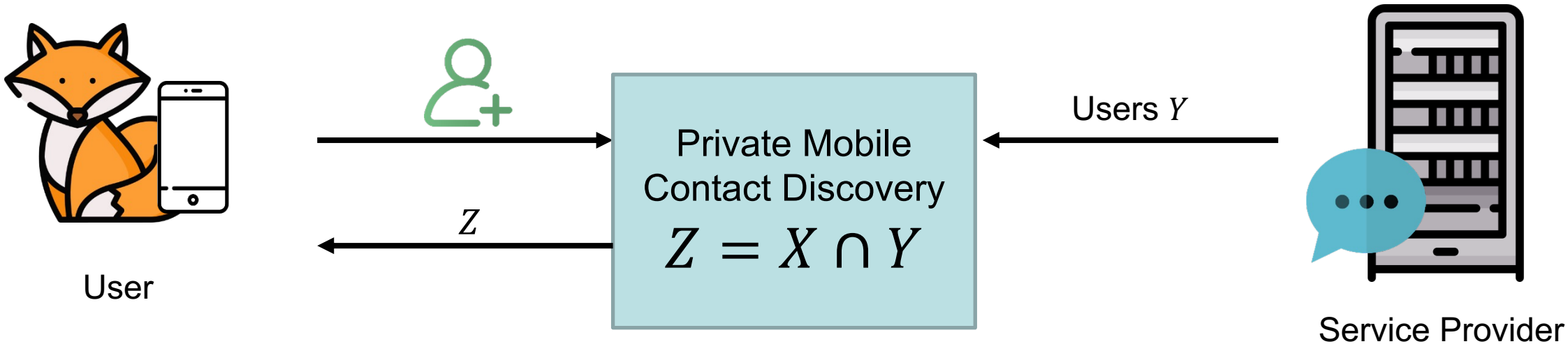
- ***Query*** every partition on each server using PIR
- Combine answers and check if tag is included

Related Work – Cuckoo Filter (CF) [FAKM14]



DATABASE UPDATES

Contact List Updates



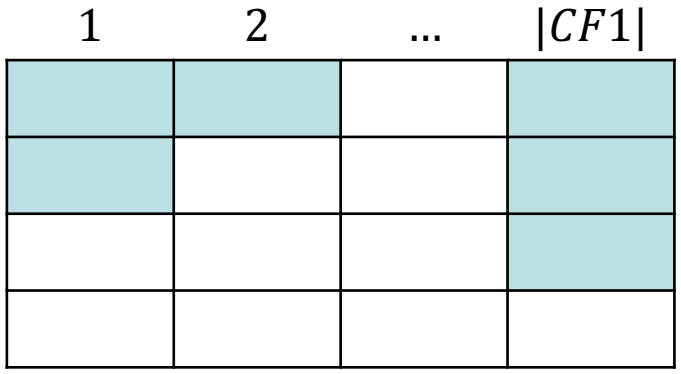
Updates – Simulation [HWS+21, HWS+23]

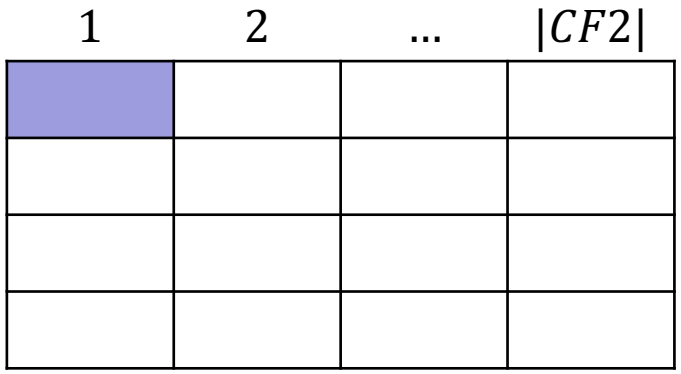
	1.5 Billion → 2 Billion Jan. 2018 – Feb. 2020	300 Million → 400 Million 2019 – 2020
Change Rate	~ 0.05% / day	~ 0.5% / day

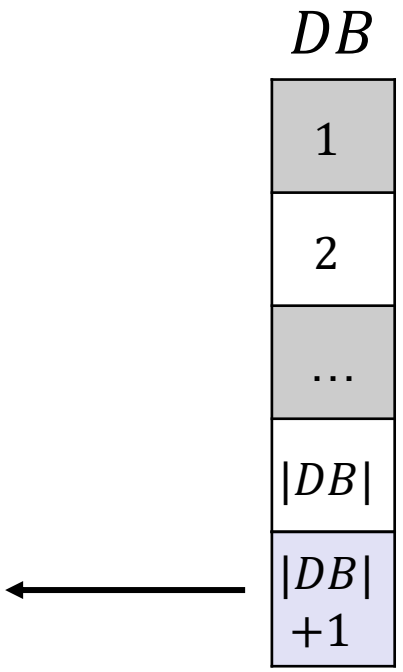
Database Updates – Additional Update Database [HWS+21, DRRT18]

Our PSI

Our PSI







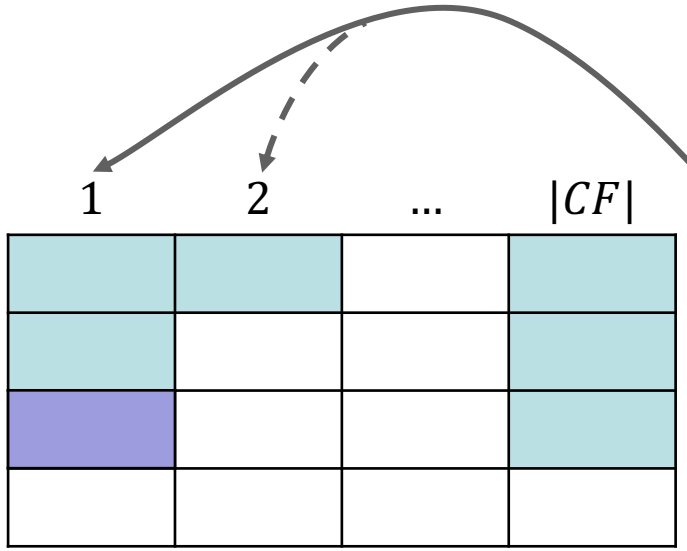
$|DB| = 2^{31}$: $0.01\ \%/day$: $2.90\ MiB/day$
$1.00\ \%/day$: $7.65\ MiB/day$

Database Updates – In-place Edits [MZRA22]

Apply Δ to $hints$

$1, \Delta_1$

Our PSI

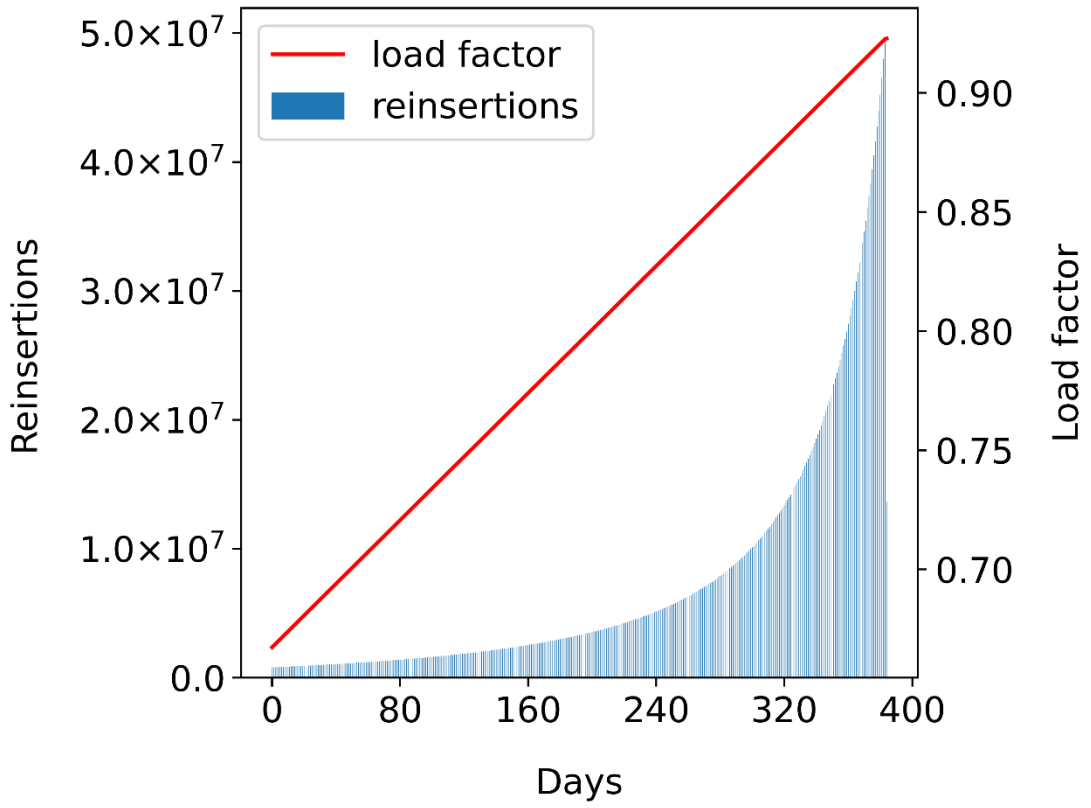


DB

1
2
...
$\|DB\|$
$\|DB\| + 1$

$$\Delta_i = CF[i]_{old} \oplus CF[i]_{new}$$

Database Updates – In-place Edits – Simulation



$|DB| = 2^{31}$:

$\mathbf{0.01}\ \%/day$: $13.52\ MiB/day$

$1.00\ \%/day$: $486.37\ MiB/day$

Implementation

Android Benchmarking Application for Mobile Private Contact Discovery [KRS+19]

https://github.com/contact-discovery/mobile_psi_android

C++ Library for Mobile Private Contact Discovery [KRS+19]

https://github.com/contact-discovery/mobile_psi_cpp

Checklist: Private Blocklist Lookups [KC21]

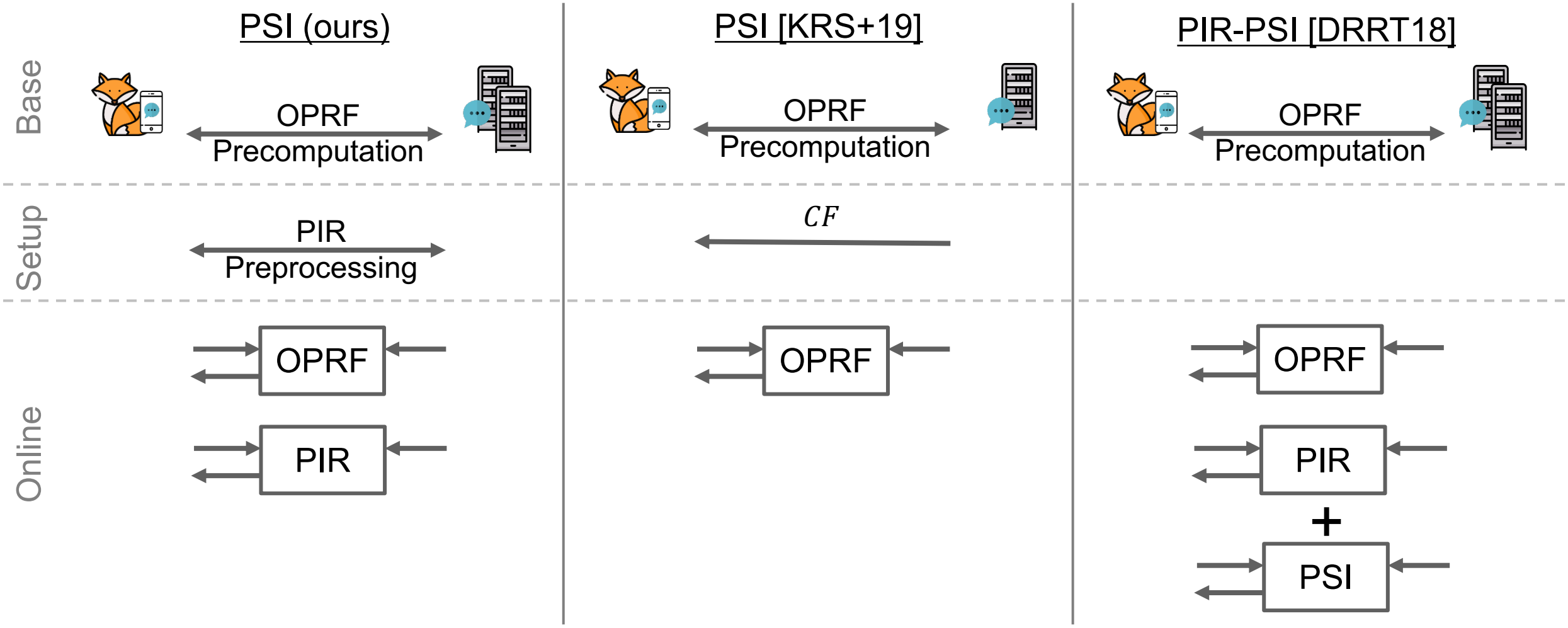
https://github.com/dimakogan/checklist

Cuckoo Filter based on [FAK14]

https://github.com/linvon/cuckoo-filter

See https://encrypto.de/code/disco

PSI Comparison – Protocol Overview



OPRF

TECHNISCHE UNIVERSITÄT DARMSTADT

Parameters		Base				Online			
OPRF Protocol	N_C	Time [s]		Comm. [MiB]		Time		Comm. [kiB]	
		WiFi	LTE	$C \rightarrow S$	$S \rightarrow C$	WiFi	LTE	$C \rightarrow S$	$S \rightarrow C$
NR-ECC [KRS+19]	1	**0.07**	**0.29**	0.03	**0.01**	0.06	0.12	0.02	4.03
	1 024	**0.15**	**0.52**	2.03	**0.01**	2.20	2.29	16.00	4 129.00
	16 384	**0.98**	**8.79**	32.03	**0.01**	34.93	34.83	256.00	66 064.00
GC-LowMC [KRS+19]	1	0.09	0.36	0.03	0.03	**0.04**	**0.07**	0.02	**2.00**
	1 024	1.26	5.39	2.03	19.53	**0.63**	**1.22**	16.00	**2 048.00**
	16 384	15.17	99.56	32.03	312.26	**10.72**	**18.62**	256.00	**32 768.00**

CS ENCRYPTO CRYPTOGRAPHY AND PRIVACY ENGINEERING

Our Protocol Results

TECHNISCHE UNIVERSITÄT DARMSTADT

Parameters				Time [s]			Comm. [MiB]	
$\|DB\|$	N_{Part}	$\|X\|$	Multi-threaded	Server Setup	Client Setup	Online	Offline	Online
2^{31}	64	2^{10}	*No*	1 988.81	961.48	6.12	264.00	2.59
			Yes	**525.09**	**286.78**	**0.74**		

< 2 seconds online time

CS ENCRYPTO CRYPTOGRAPHY AND PRIVACY ENGINEERING

Summary – Evaluation

TECHNISCHE UNIVERSITÄT DARMSTADT

Asymptotic Communication Complexity

	[KRS+19]	Our Protocol
Base	$\lvert X\rvert$	$\lvert X\rvert$
Setup	$\lvert DB\rvert$	$N_{Part}\sqrt{\lvert CF_p\rvert}$
Online	$\lvert X\rvert$	$\log\lvert CF_p\rvert\,(\lvert X\rvert + \sqrt{\lvert X\rvert N_{Part}\log N_{Part}})$

8× Larger set sizes

31× Reduced offline communication

< 2 sec. online time
for $\lvert X\rvert = 2^{10}$ and $\lvert DB\rvert = 2^{31}$

$\lvert X\rvert$	Number of client inputs, e.g., 2^{10}
$\lvert DB\rvert$	Number of server inputs to PSI Protocol, e.g., 2^{31}
$\lvert CF_p\rvert$	Number of server inputs to PIR Protocol, number of CF buckets per partition, e.g.,2^{24}
N_{Part}	Number of CF Partitions, e.g., 32

CS ENCRYPTO CRYPTOGRAPHY AND PRIVACY ENGINEERING