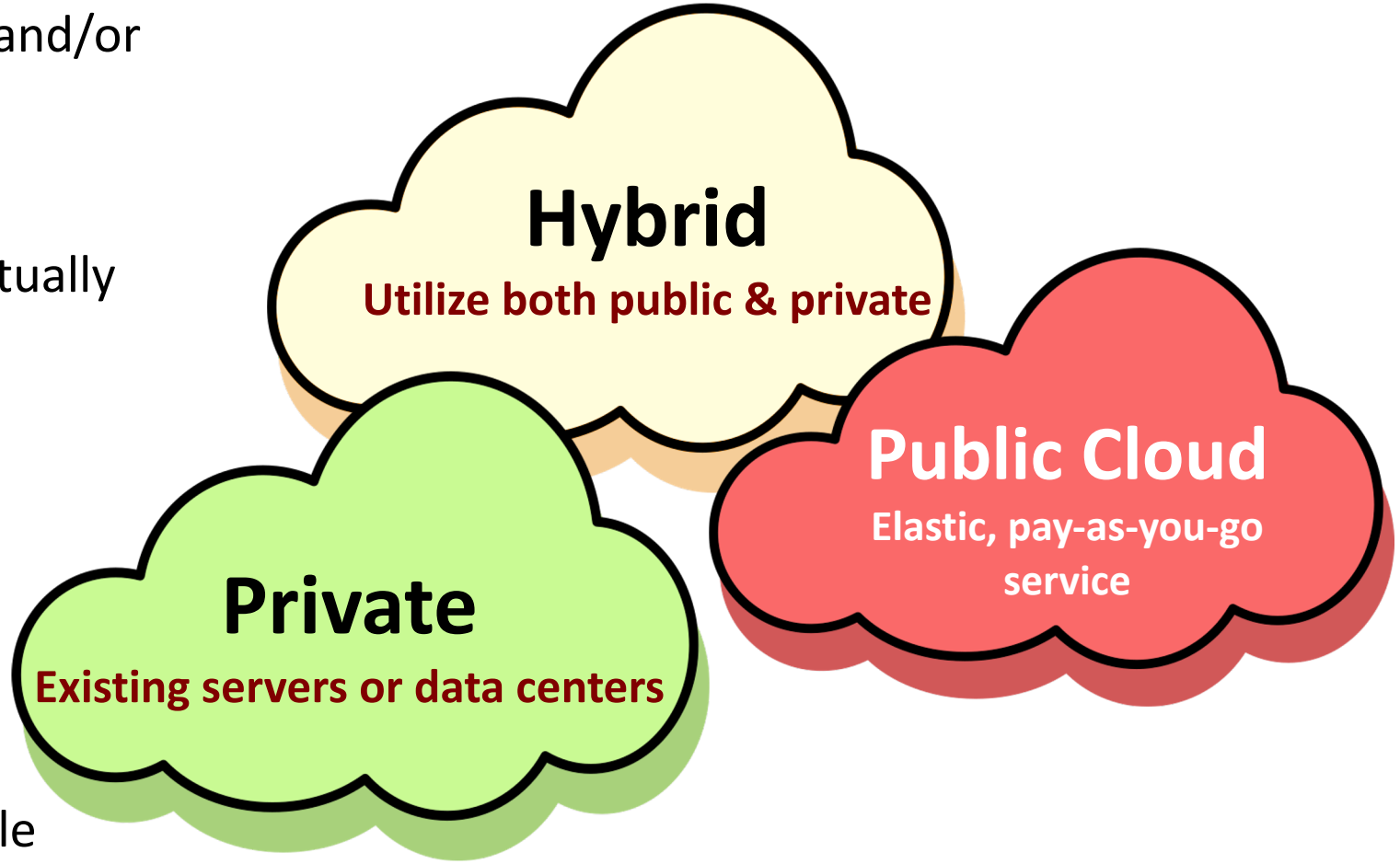
Secure Data Outsourcing Over the Years

Sharad Mehrotra
University of California, Irvine, USA.

Slides derived from a recent tutorial at ICDE 2020 by Shantanu Sharma, Anton Burtsev, and Sharad Mehrotra

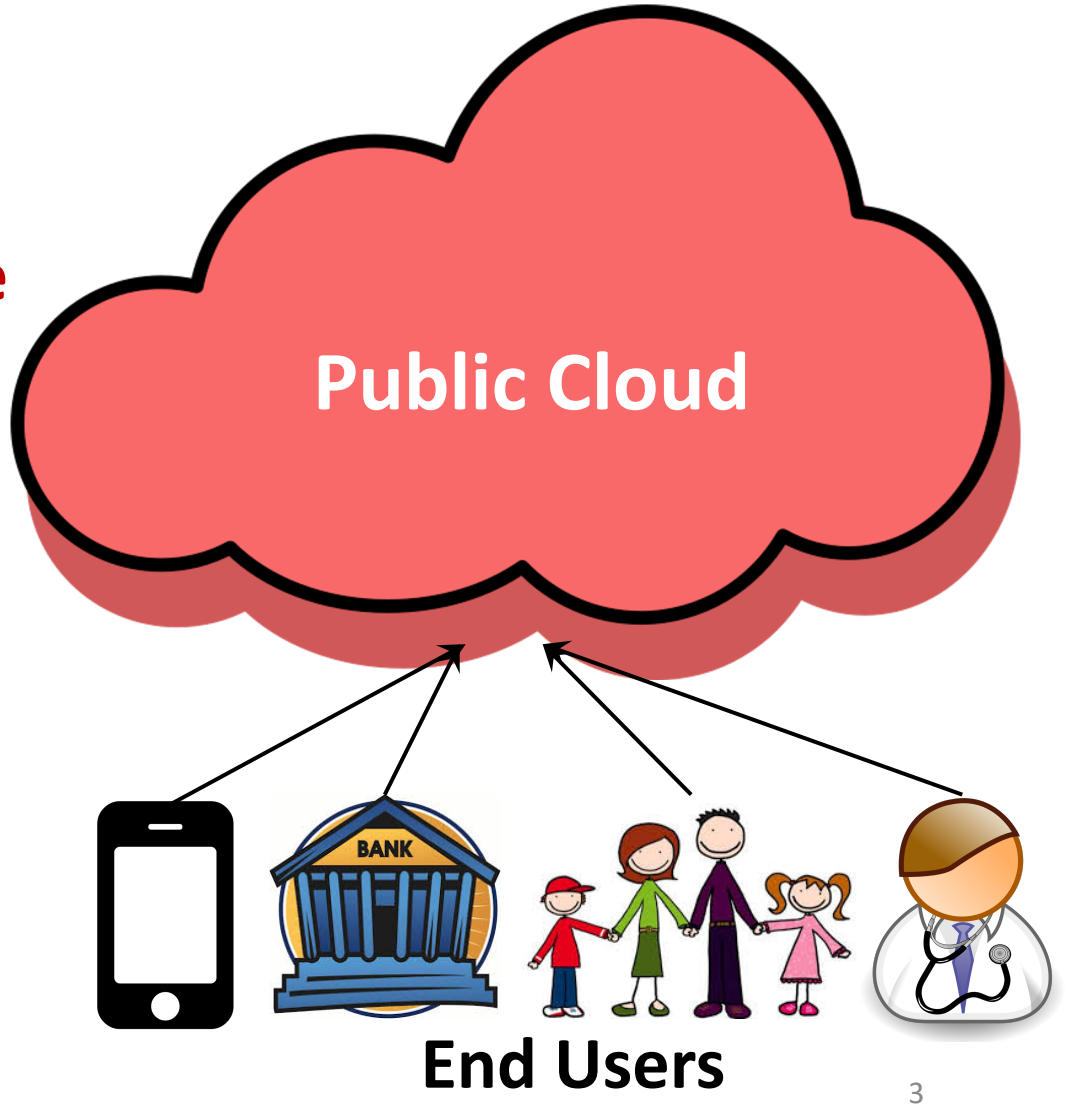
Cloud Computing

- **Utility model**
  - Pay for only what you use
  - No infrastructure build-up cost and/or database administration costs
- **Elastic**
  - Use as much as your needs (virtually limitless)
- **No system management headaches**
  - Failure, loss of data, software upgrades, patches, bug fixes
- **Cost amortization**
  - Cheaper due to economy of scale
  - Better control over IT investment



Key Challenge: Loss of Control

- Data resides in **shared systems** administration of which is not in owners' control
- Unknown applications and processes **share resources** with apps and data.
- Data owners have **no control over the cloud**'s internal data security personnel, policies or their enforcement
  - Insider attacks
  - Data mining attacks leading to information leakage
  - Cloud providers compliance to government subpoenas



Adversarial Cloud Model

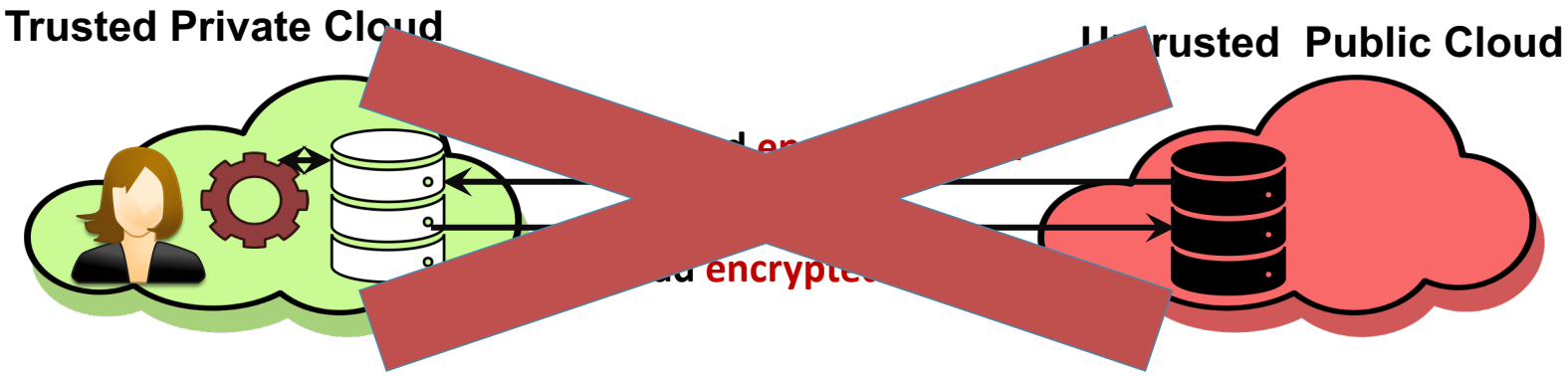
- **Honest-but-Curious versus Malicious adversary**
  - **Honest-but curious**
    - Executes protocols correctly, but wishes to learn about data
  - **Malicious**
    - Might sabotage data or computation
- **Passive versus Active Adversary**
  - **Passive**
    - Makes inferences based on passive observations – ciphertext, queries, workload, and access patterns
  - **Active**
    - May actively inject new data, execute queries, or interfere with the execution

What is The Solution?

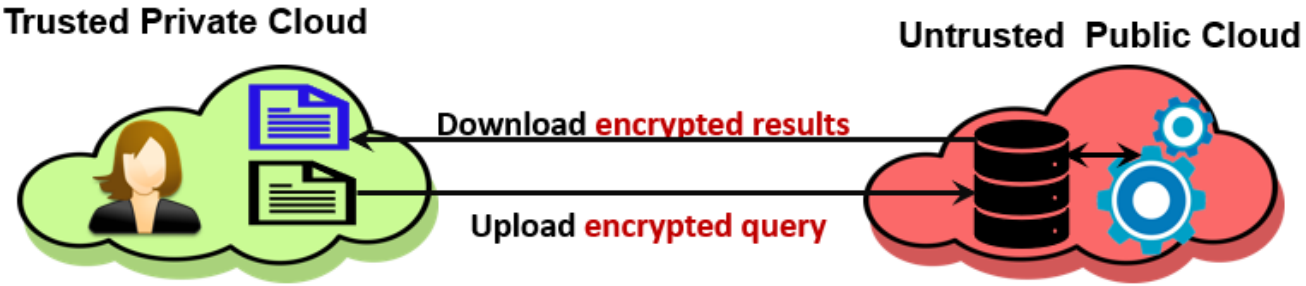
Encrypt sensitive data before uploading to the cloud

Secure Computing

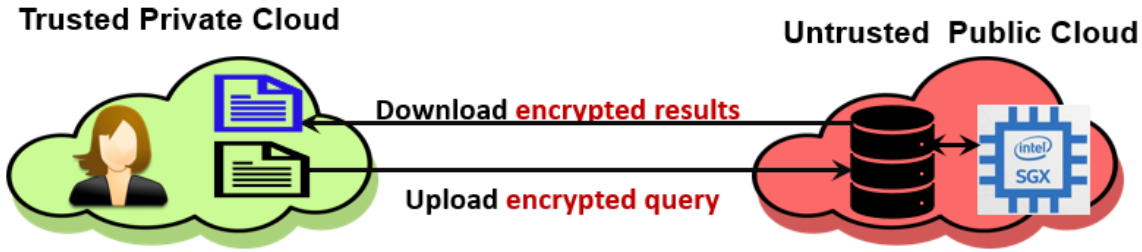
Download the encrypted data and compute at the trusted side



Cryptographic Solutions at the Cloud

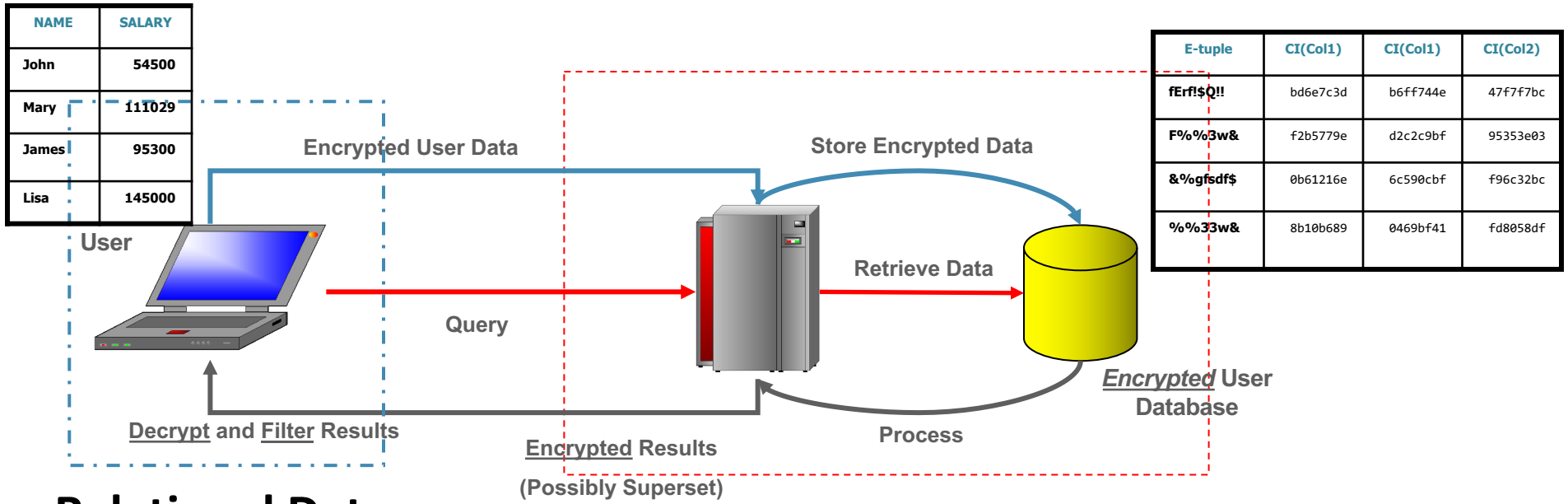


Exploiting Trusted Computing



Data Processing over Encrypted Data

Executing SQL over Outsourced Encrypted Data



- **Encrypting Relational Data**
  - ***Store cipher-indices*** in the form of additional columns in relations
- **Processing SQL queries over encrypted relations**
  - Compute as much as possible over encrypted domain using cipher-indices (***encrypted query processing***)
  - Transfer intermediate results to client, decrypt and execute rest of the query at the client. (***partitioned execution***)

Hacigumus, Iyer, Li, Mehrotra, Executing SQL over encrypted data… SIGMOD 2002

Homomorphic Encryption

- Fully homomorphic approach
  - Very inefficient and not practical
- Partially homomorphic
  - Additive: e.g., Paillers
  - Multiplicative: e.g., Elgamal
- Searchable encryption
  - Enables comparisons in ciphertext without decryption

Common Attacks

- **An adversary may learn about data:**
  - From ciphertext (**ciphertext representation-based attack, e.g., order of values**)
  - From prior knowledge of data distribution (**frequency-count attack**)
  - From knowledge of frequency of queries (**workload-skew attack**)
  - From the size of the output to a query (**output-size attack**)
  - From the access pattern used by the mechanism in answering a query (**access-pattern attack**)
  - From knowledge of queries that have executed (**search-pattern attack**)

Mix with adversarial background knowledge → Reveal secured data

Encrypted Data Processing over the Years

- **Encryption-based Techniques**
  - Bucketization [Hore et al. VLDB 04]
  - Searchable Encryption [Song et al., IEEE SP 00]
  - Secure indexes – encrypted Bloom filters [Goh, 03]
  - Bilinear maps [Boneh et al., EuroCrypt 03]
  - Order-Preserving Encryption (OPE) [Agrawal et al., SIGMOD 04]
  - Modular-OPE [Boldyreva et al., CRYPTO 11]
  - Conjunctive keyword search [Golle et al., ACNS 04]
  - Encrypted inverted lists [Curtmola et al., CCS 06]
  - Fully homomorphic encryption [Gentry, STOC 09]
  - Onion encryption [Popa et al., SOSP 11]
  - Dynamic Searchable Encryption [Cash et al.NDSS 14]
  - PBTree [Li et al., VLDB 14]
  - IBTree [Li et al., ICDE 17]
- **Secret-Sharing Techniques**
  - Shamir's secret-sharing [Shamir, CACM 79]
  - Multi-Linear Secret-Sharing Schemes [Brickell et al., J. of Cryptology 91, Bertilsson et al., AUSCRYPT 92]
  - Verifiable secret sharing [Rabin et al., STOC 89]
  - Proactive Secret Sharing [Herzberg et al., CRYPTO 95]
  - Function Secret Sharing [Boyle et al., EUROCRYPT 15]
  - Homomorphic secret sharing [Boyle et al. CRYPTO 16]
  - Accumulating Automata [Dolev et al., TCS 19]

Inference Attacks on Property-Preserving Encrypted Databases

Muhammad Naveed
UIUC*
naveed2@illinois.edu

Seny Kamara
Microsoft Research
senyk@microsoft.com

Charles V. Wright
Portland State University
cvwright@cs.pdx.edu

ABSTRACT

Many encrypted database (EDB) systems have been proposed in the last few years as cloud computing has grown in popularity and data breaches have increased. The state-of-the-art EDB systems for relational databases can handle SQL queries over encrypted data and are competitive with commercial database systems. These systems, most of which are based on the design of CryptDB (*SOSP 2011*), achieve these properties by making use of property-preserving encryption schemes such as deterministic (DTE) and order-preserving encryption (OPE).

In this paper, we study the concrete security provided by such systems. We present a series of attacks that recover the plaintext from DTE- and OPE-encrypted database columns using only the encrypted column and publicly-available auxiliary information. We consider well-known attacks, including frequency analysis and sorting, as well as new attacks based on combinatorial optimization.

We evaluate these attacks empirically in an electronic medical records (EMR) scenario using real patient data from

General Terms

Security, Experimentation

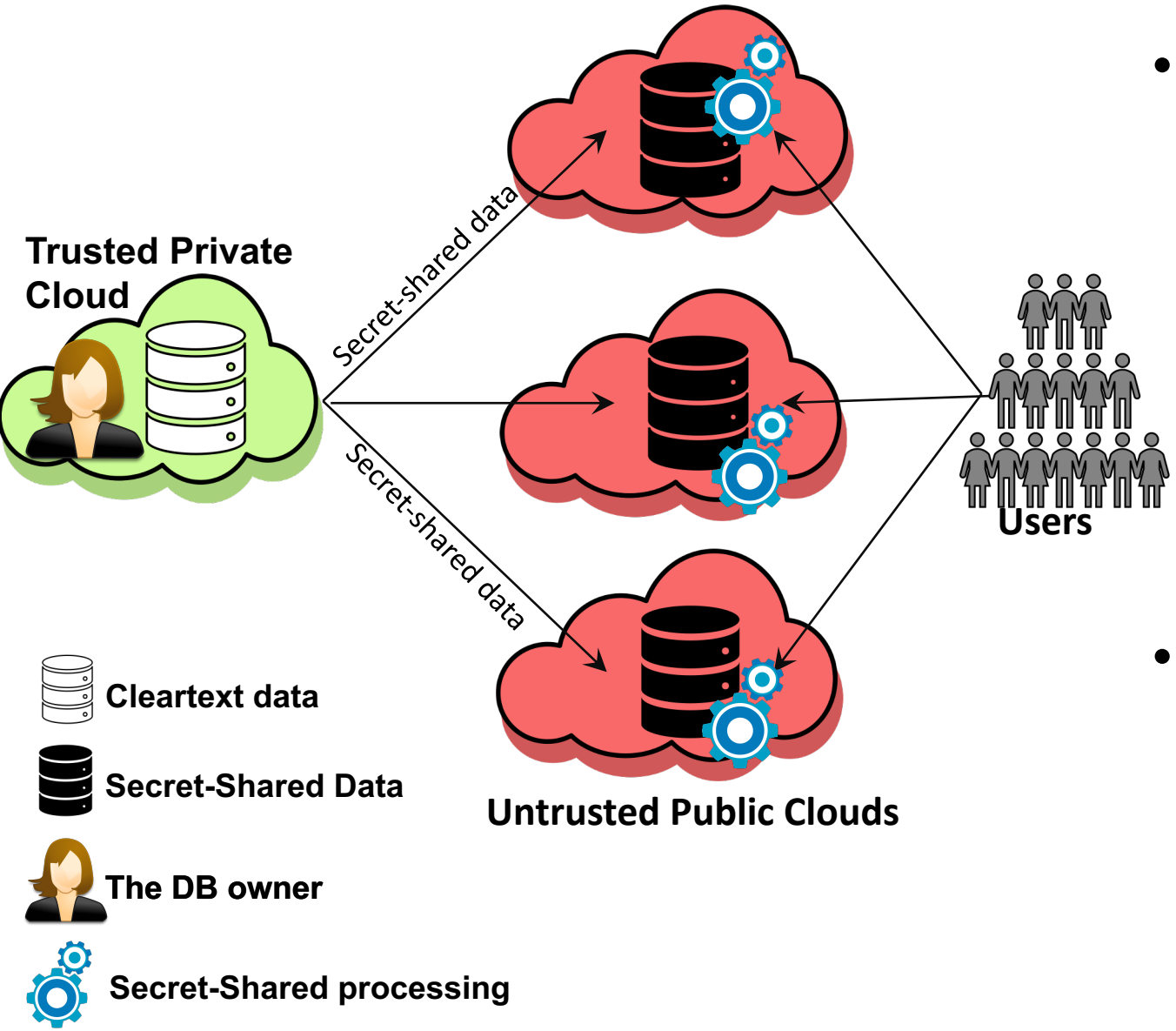
Keywords

inference attacks; encrypted databases; property-preserving encryption; deterministic encryption; order-preserving encryption

1. INTRODUCTION

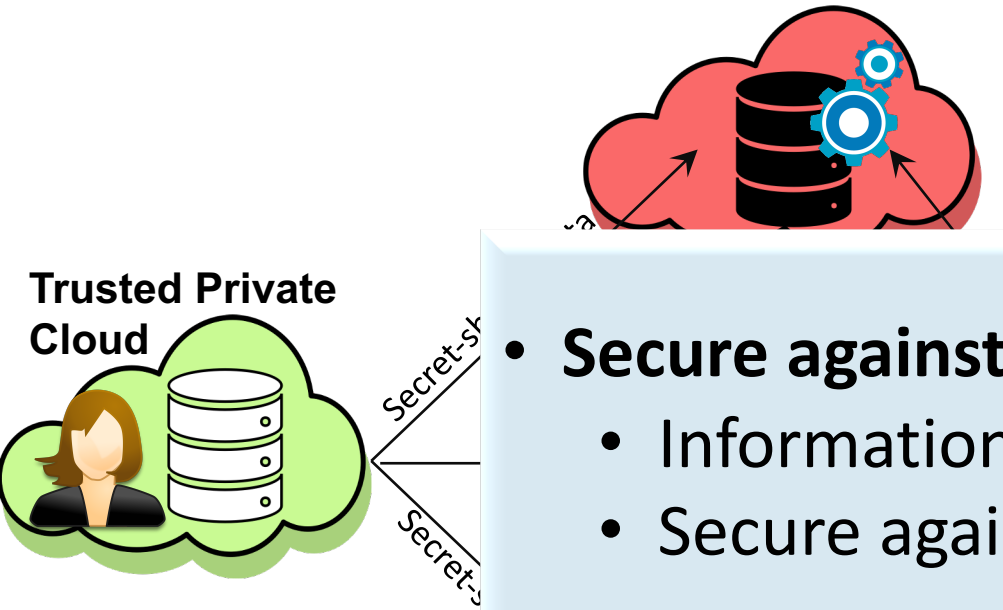
As an increasing amount of private data is being collected and stored by corporations and governments, database security has become a critical area in both research and industry. High-profile data breaches like the Anthem breach in which a database of 80 million healthcare records was compromised or the Community Health Systems breach in which 4.5 million HIPAA protected (non-medical) records were stolen have fueled interest in database encryption techniques

MPC and Secret Shared Mechanisms



- **Techniques:**
  - Secret-sharing [Shamir, CACM, 1979]
  - Distributed Point Function [Gilboa et al., EUROCRYPT, 2014.]
  - Function secret-sharing [Boyle et al., EUROCRYPT, 2015]
  - Homomorphic Secret-Sharing [Boyle et al., CCS, 2017]
  - Accumulating-Automata [Dolev et al, SCC@ASIACCS , 2014]
  - OBSCURE [Gupta et al, CS@UCI, 2019]
  - Conclave [Volgushev et al. arxiv, 2019]
  - SMCQL [Bater et al., PVLDB, 2017]
- **Systems:**
  - Jana by Galois
  - Partisia
  - PULSAR by Stealth Software Technologies
  - Secret Double Octopus and SecretSkyDB Ltd
  - Sharemind by Cybernetica
  - Unbound Tech.

MPC and Secret Shared Mechanisms



- **Secure against stronger adversaries**
  - Information-theoretically secure
  - Secure against access-pattern-based attacks
- **However, much more expensive**
  - 5-6 order of magnitude expensive compared to plain text processing


Cleartext data

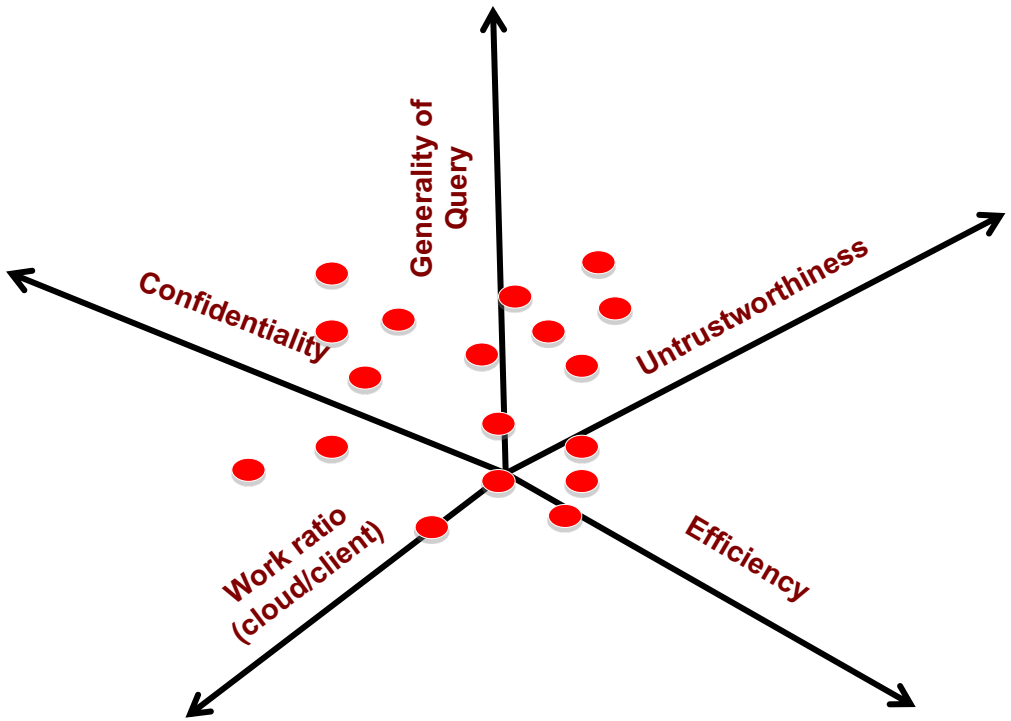

Secret-Shared Data


The DB owner


Secret-Shared processing

- **Techniques:**
  - Secret-sharing [Shamir, CACM, 1979]
  - Distributed Point Function [Gilboa et al., EUROCRYPT,
  - t al., EUROCRYPT, 2015]
  - oyle et al., CCS, 2017]
  - et al, SCC@ASIACCS ,
  - 2019]
  - 2019]
  - 7]
  - hnologies
  - Secret Double Octopus and SecretSkyDB Ltd
  - Sharemind by Cybernetica
  - Unbound Tech.

Cryptographic Solution Landscape



- **Large number of solutions**
  - **Represent points in the spectrum of possibilities**

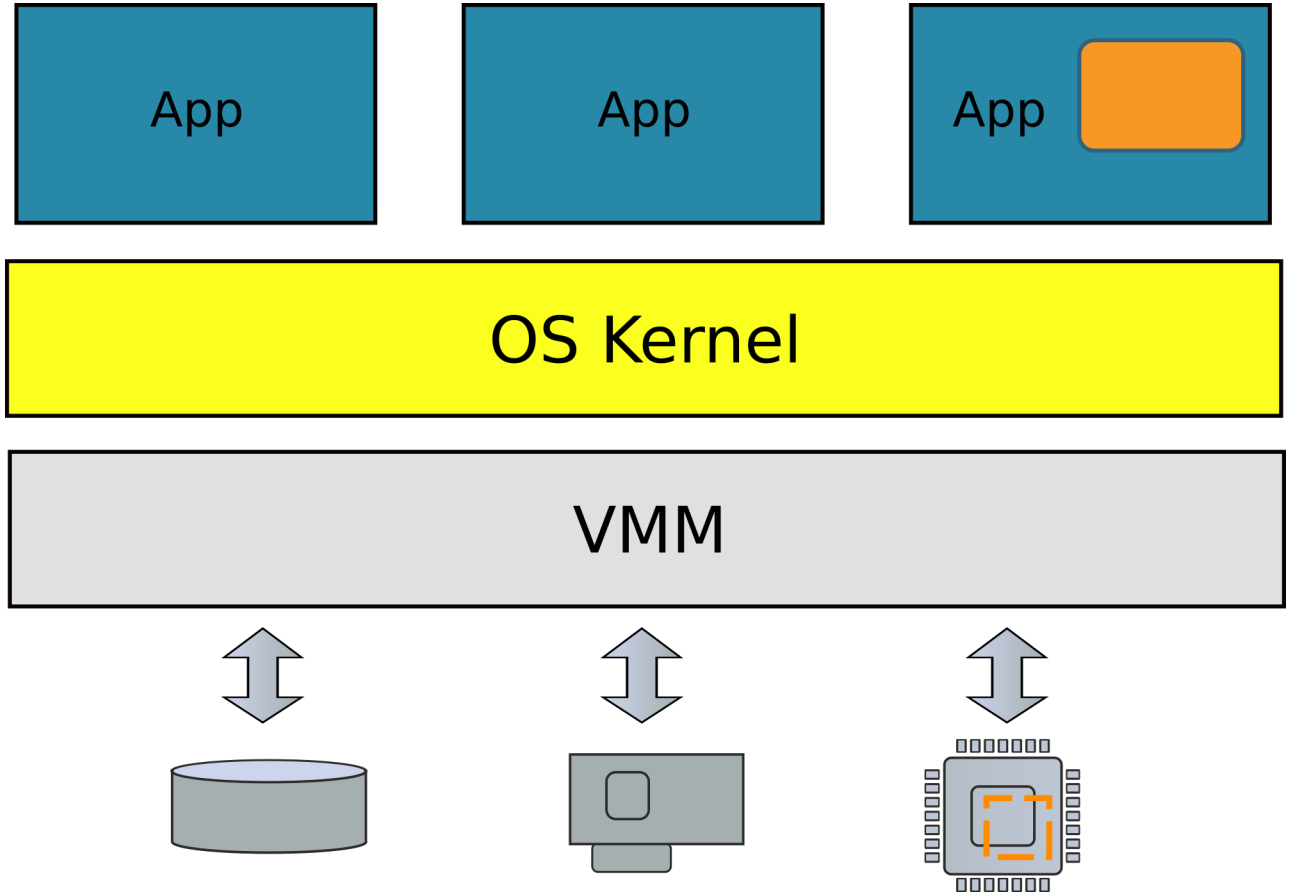
- **Explore different tradeoffs**
  - **Efficiency –** overhead, indexable?
  - **Generality -** What queries can the technique support – selection, range, join, aggregation
  - **Dynamic Operations -** Does the scheme support insertion/deletions/updates?
  - **Client-Side Execution -** How much work does the client have to do? During insertion/ updates/ queries.
  - **Security -** How much security does the scheme offer? Quantifiable leakage, e.g., orderability, distribution? Semantic security?

Exploiting Trusted Computing Platforms

Secure hardware

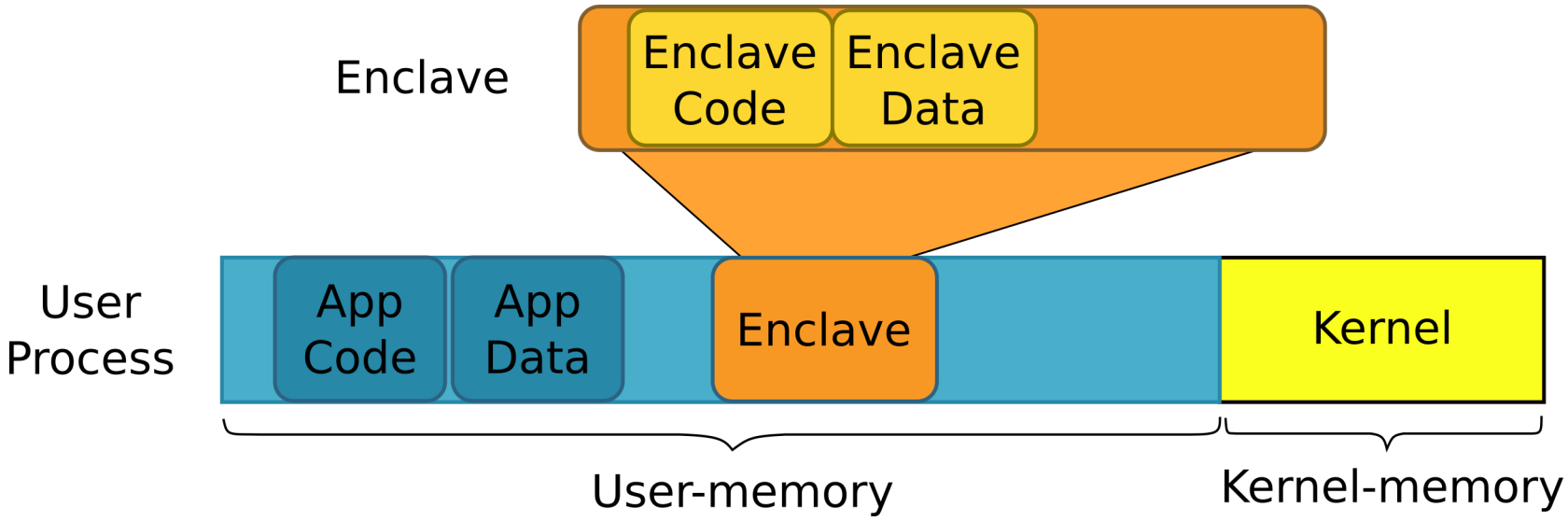
Enclaves

- Applications can protect their secrets
- TCB is small
  - Intel CPU
  - App code itself
- Protected from malicious
  - BIOS
  - SMM
  - Hypervisor
  - Kernel
- Familiar application environment

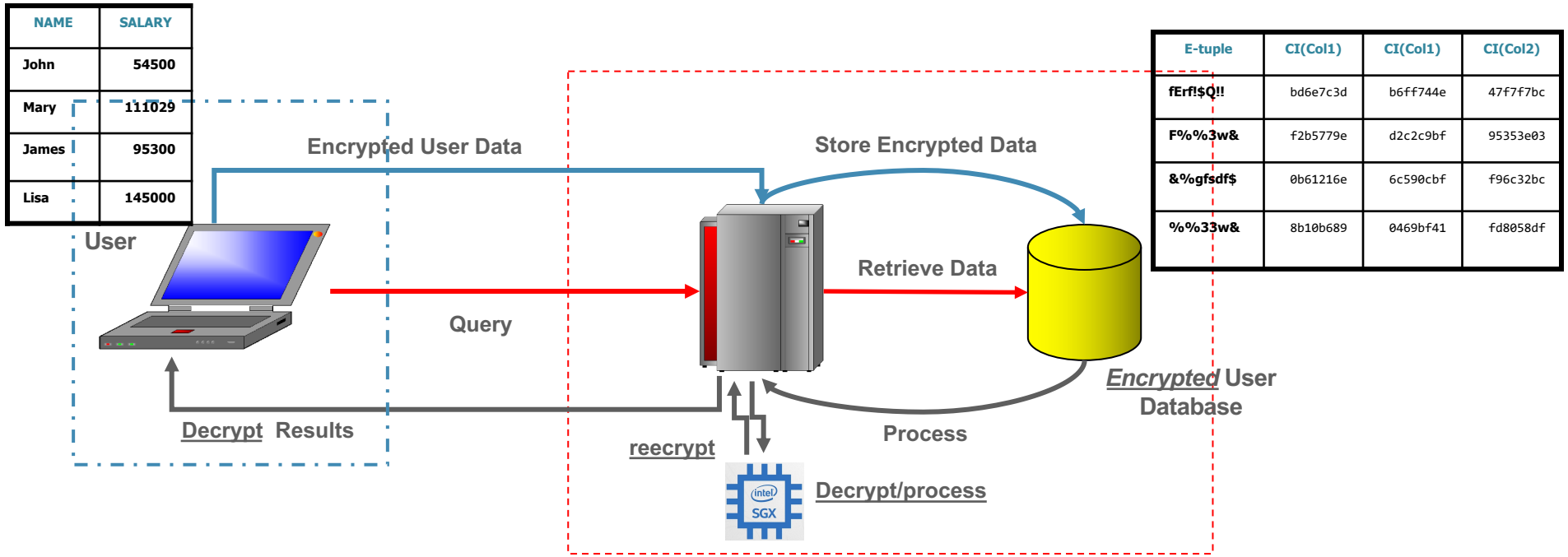


SGX Enclaves

- Trusted execution environment embedded in the process
  - It's own code and data
  - Controlled entry points
  - Multi-threading
- Confidentiality
- Integrity



Executing SQL using Trusted Hardware



- **Secure hardware at the cloud acts as a trusted agent of the data provider**
- **Queries executed collaboratively between trusted hardware and the untrusted server**
- **Secure FPGA-based solutions (e.g.,** Microsoft Cipherbase)
- **Intel SGX-based solutions (e.g.,** Opaque, EnclaveDB, VC3, HardIDX)

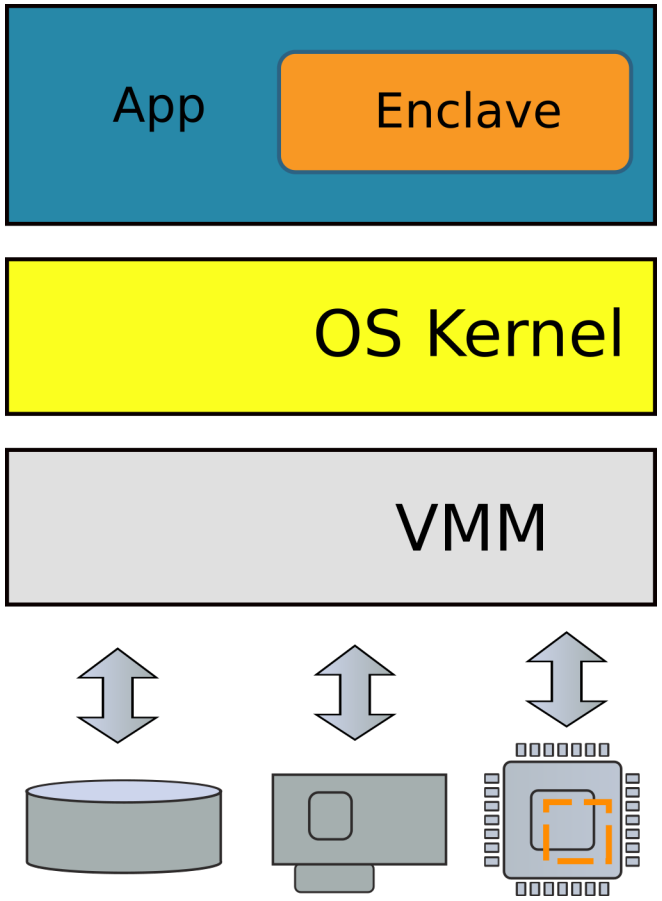
Performance

- Enters and exits are expensive
  - EEXIT 3,330 cycles
  - EENTER 3,800 cycles
  - Intel SDK adds another 800 cycles
  - Normal syscall is 250 cycles
- Memory is encrypted
- Limited physical memory
  - 128MB (in practice only 90MB available for your application)
  - 40,000 cycles per enclave page cache (EPC) fault (25K driver, 7K exit/entry, 8K indirect)
- ***10-33X slowdown for simple key-value stores***

Reference: Orenbach, et al. "Eleos: ExitLess OS services for SGX enclaves." EuroSys'17.

Security

- Powerful Adversarial Model -- OS + VMM
  - Controlled execution environment
  - Control over page faults
  - Suspending execution
    - Single stepping
  - Flushing caches



SGX-Step: A Practical Attack Framework for Precise Enclave Execution Control

Jo Van Bulck
imec-DistriNet, KU Leuven
jo.vanbulck@cs.kuleuven.be

Frank Piessens
imec-DistriNet, KU Leuven
frank.piessens@cs.kuleuven.be

Raoul Strackx
imec-DistriNet, KU Leuven
raoul.strackx@cs.kuleuven.be

Abstract

Protected module architectures such as Intel SGX hold the promise of protecting sensitive computations from a potentially compromised operating system. Recent research convincingly demonstrated, however, that SGX's strengthened adversary model also gives rise to to a new class of powerful, low-noise side-channel attacks leveraging first-rate control over hardware. These attacks commonly rely on frequent enclave preemptions to obtain fine-grained side-channel observations. A maximal temporal resolution is achieved when the victim state is measured after every instruction. Current state-of-the-art enclave execution control schemes, however, do not generally achieve such instruction-level granularity.

This paper presents SGX-Step, an open-source Linux kernel framework that allows an untrusted host process to configure APIC timer interrupts and track page table entries directly from user space. We contribute and evaluate an improved approach to single-step enclaved execution at instruction-level granularity, and we show how SGX-Step enables several new or improved attacks. Finally, we discuss its

concerns, the past years have seen a significant research effort [3, 6, 9] on Protected Module Architectures (PMAs) that support isolated execution of security-sensitive application components or *enclaves* with a minimal Trusted Computing Base (TCB). These proposals have in common that they enforce security primitives directly in hardware, or in a small hypervisor, so as to prevent the untrusted OS from accessing enclaved code or data directly, while still leaving it in charge of shared platform resources such as system memory or CPU time. With the arrival of Intel's Software Guard eXtensions (SGX) [6, 7], such strong hardware-enforced trusted computing guarantees are now available on mainstream consumer devices.

Recent research demonstrated, however, that the increased capabilities of a privileged PMA attacker allow her to construct high-resolution, low-noise channels to spy on enclaved execution. Specifically, the past months have seen a steady stream of kernel-level SGX attacks exploiting information leakage from page tables [13, 15], CPU caches [4, 10], or branch prediction units [8]. These attacks commonly exploit

Side-Channel Attacks

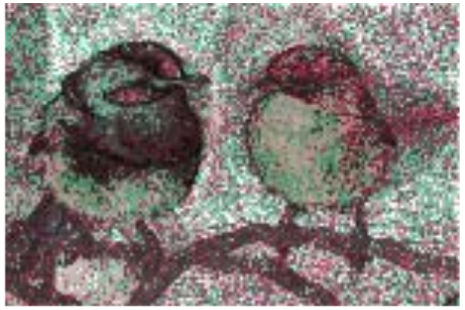
- Every architectural component of the CPU
  - Branch target buffers
    - S. Lee et al., “Inferring fine-grained control flow inside SGX enclaves with branch shadowing,” in USENIX Security, 2017
    - G. Chen et al., “SgxPectre attacks: Stealing intel secrets from SGX enclaves via speculative execution,” arXiv preprint, 2018.
  - Pattern-history table
    - D. O'Keeffe et al., "Spectre attack against SGX enclave," 2018
  - Caches
    - Brasser et al., "Software grand exposure: SGX cache attacks are practical," in WOOT, 2017
    - J. Gotzfried et al., "Cache attacks on Intel SGX," in EuroSec, 2017
    - A. Moghimi et al., "Cachezoom: How SGX amplifies the power of cache attacks," in CHES, 2017
    - M. Hahnel et al., "High-resolution side channels for untrusted operating systems," in USENIX ATC, 2017
    - M. Schwarz et al., "Malware guard extension: Using SGX to conceal cache attacks," in DIMVA, 2017
  - DRAM row buffer
    - W. Wang et al., "Leaky cauldron on the dark land: Understanding memory side-channel hazards in SGX," in CCS, 2017
  - Page-tables
    - W. Wang et al., “Leaky cauldron on the dark land: Understanding memory side-channel hazards in SGX,” in CCS, 2017
    - J. Van Bulck et al., “Telling your secrets without page faults: stealthy page table-based attacks on enclaved execution,” in USENIX, 2017
  - Page-fault exception handlers
    - Y. Xu et al., “Controlled-channel attacks: Deterministic side channels for untrusted operating systems,” 2015
    - S. Shinde and other, “Preventing page faults from telling your secrets,” in CCS, 2016
  - Speculative execution
    - J. V. Bulck et al., “Foreshadow: Extracting the keys to the Intel SGX kingdom with transient out-of-order execution,” in USENIX, 2018

Example: Recovering JPEG Images

```
GLOBAL(void) jpeg_idct_islow (j_decompress_ptr cinfo,
  jpeg_component_info * compptr, JCOEFPTR coef_block,
  JSAMPARRAY output_buf, JDIMENSION output_col)
{
  ...
  /* Pass 1: process columns from input... */
  inptr = coef_block;
  quantptr = (ISLOW_MULT_TYPE *) compptr->dct_table;
  wsptr = workspace;
  for (ctr = DCTSIZE; ctr > 0; ctr--) {
    /* Due to quantization, we will usually find that
     * many of the input coefficients are zero,
     * especially the AC terms.  We can exploit this
     * by short-circuiting the IDCT calculation for any
     * column in which all the AC terms are zero. In
     * that case each output is equal to the DC
     * coefficient (with scale factor as needed). With
     * typical images and quantization tables, half or
     * more of the column DCT calculations can be
     * simplified this way.
     */
    if (inptr[DCTSIZE*1]==0 && inptr[DCTSIZE*2]==0 &&
          inptr[DCTSIZE*3]==0 && inptr[DCTSIZE*4]==0 &&
          inptr[DCTSIZE*5]==0 && inptr[DCTSIZE*6]==0 &&
          inptr[DCTSIZE*7]==0) {
      /* AC terms all zero */
      ... SIMPLE COMPUTATION ...
      inptr++; quantptr++; wsptr++;
      continue;
    }
    ... COMPLEX COMPUTATION ...
    inptr++; quantptr++; wsptr++;
  }
```

- JPEG
  - Process 8x8 blocks
  - Function fits on one page
    - Cannot reason about input-dependent page-faults
- Can reason about number of page-faults
  - Optimizations in the code take shortcuts

Reference: Y. Xu et al., “Controlled-channel attacks: Deterministic side channels for untrusted operating systems,” 2015

Original	Recovered	Original	Recovered
			
			

Today, SGX does not offer much security…..

Research to protect against vulnerabilities ongoing

Some problems will be fixed in hardware – cache, branch prediction

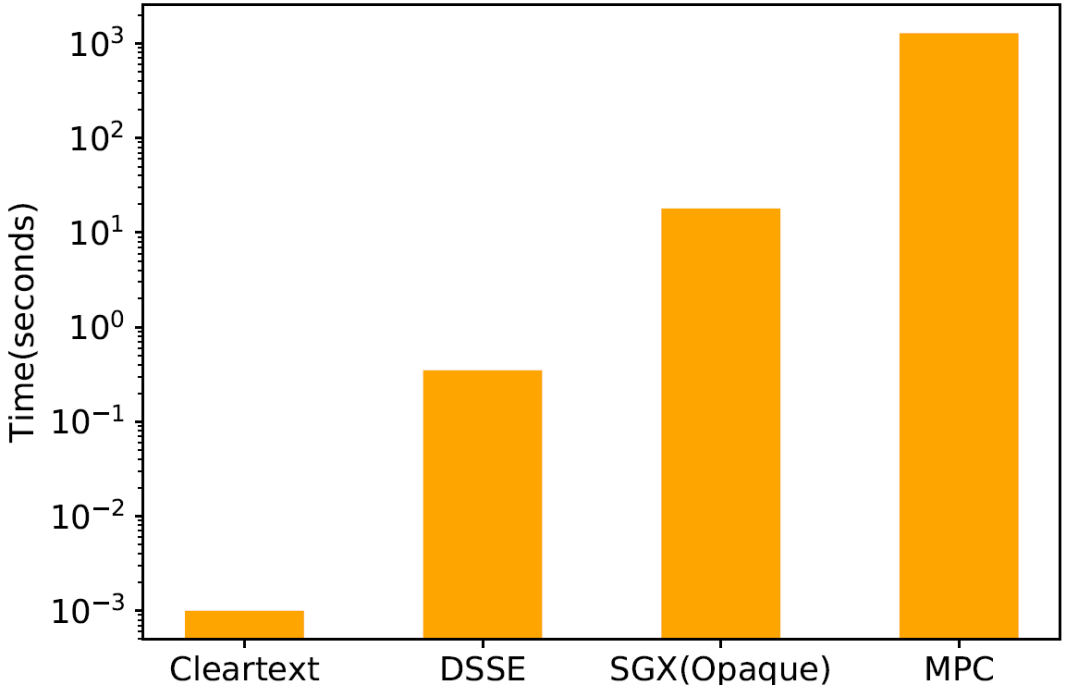
Some problems are too hard to fix completely – will result in unacceptable overheads (e.g., enclave memory access patterns)

Bottomline

- **Despite 20 years of progress in cryptography, secure hardware, and secure data processing.**

Computation Cost & Security

Selecting 8 rows from TPC-H LineItem table of 1M rows and 7 columns
(orderkey, linestatus, quantity, partkey, suppkey, linenumber, returnflag)
(8 core 3.5 GHz 32 GB machine)



- **Cryptographic Overheads:**
  - Searchable encryption – ~2 orders of magnitude
  - Secure hardware - ~3-4 order of magnitude
  - MPC based solution - ~5-6 orders of magnitude

Security Threats

	Resilient to attacks			
Techniques	**Data at rest**	**During query execution**		
	Ciphertext indistinguishability	**Output-Size**	**Workload-skew**	**Access-patterns**
Full Download	✓	✓	✓	✓
Deterministic Encryption/OPE	X	X	X	X
Non-Deterministic Encryption	✓	X	X	X
Searchable encryption	✓	X	X	X
Homomorphic + ORAM	✓	X	X	✓
Shamir's Secret-sharing	✓	X	X	✓
Multi-party computations-Jana	✓	X	X	✓

✓ **represents technique is resilient to a given attack.**

Reference: Sharad Mehrotra, Shantanu Sharma, and Jeffrey D. Ullman. "Scaling Cryptographic Techniques by Exploiting Data Sensitivity at a Public Cloud." In *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy*, pp. 165-167. ACM, 2019.

That has not stopped the industry or academia …

- **Encryption-based Systems**
  - CryptDB [Popa et al., SOSP 11]
  - Monomi [Tu et al.. VLDB 13]
  - Cipherbase [Arasu et al., CIDR 13]
  - TrustedDB [Bajaj et al., IEEE TKDE 13]
  - CorrectDB [Bajaj et al., VLDB 13]
  - ZeroDB [Egorov et al., arxiv 16]
  - MrCrypt [Tetali et al., OOPSLA 13]
  - EncKB [Yuan et al., ASIACCS 17]
  - Microsoft Always Encrypted
  - Oracle 12c
  - Amazon Aurora
  - MariaDB

- **Secret-Sharing-based Systems**
  - SSSDB [Avni et al., ALGOCLOUD 15]
  - Splinter [Wang et al., NSDI 17]
  - OBSCURE [Gupta et al, VLDB 19]
  - Cybernetica
  - Jana by Galois Inc.
  - Partisia
  - Secret Double Octopus
  - SecretSkyDB Ltd
  - PULSAR by Stealth Software Technologies Inc.
  - Unbound Tech.

a testimony to importance of problem

Academic/industrial papers

Industrial/startup systems

Key Question…

Can we design an outsourcing solution for that is simultaneously??

Efficient – *significantly better compared to downloading cryptographically secured data*, and

Secure – *similar to downloading the data and local processing*

Reference: Sharad Mehrotra, Shantanu Sharma, Jeffrey Ullman, and Anurag Mishra. "Partitioned data security on outsourced sensitive and non-sensitive data." In 2019 IEEE ICDE, pp. 650-661. IEEE, 2019.

A Way forward …right sizing the solution to the problem

- **Traditionally, secure data processing considers a world to be binary…**
  - Either all data needs to be protected, or
  - No data needs protection

- **In contrast, the real-world might be a lot grayer….**
  - Organization data is often only partially sensitive
    - Sensitivity dictated by policies
  - Why pay cryptographic overheads to strongly protect all the data, when only a small portion needs to be protected?
  - Commercial systems (e.g., Jana by Galois Inc.) are beginning to explore such solutions

- ***Work on understanding the security and performance implications of using multiple cryptographic techniques simultaneously is just beginning.***

"Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Networking and Information Technology Research and Development Program."

The Networking and Information Technology Research and Development (NITRD) Program

Mailing Address: NCO/NITRD, 2415 Eisenhower Avenue, Alexandria, VA 22314

Physical Address: 490 L'Enfant Plaza SW, Suite 8001, Washington, DC 20024, USA Tel: 202-459-9674, Fax: 202-459-9673, Email: nco@nitrd.gov, Website: https://www.nitrd.gov