Privacy and Integrity Preserving Computations with CRISP

S. Chatel, A. Pyrgelis, J.R. Troncoso-Pastoriza, J-P. Hubaux
Laboratory for Data Security, EPFL
USENIX Security 2021
Data Analysis Cycle

**Data source** creates data about a user

The **user** wishes to obtain a service from a provider

A **provider** is interested in gaining some information

This data flow models several real-life use case:

- Medical and Health Programs
- Loyalty programs
- Insurance Toll
- Activity Tracking
- Smart Metering
Motivation

Real-world data flows are vulnerable to attacks on privacy and authenticity e.g.:

- **Smart Metering**
  - Load balancing, forecast, energy savings
  - Inference on user’s behavior
  - Protection required to avoid disruption

- **Personal Tracking**
  - Location-based activity tracking
  - Privacy breach about user’s behaviour or even national security interests
  - Protection required for insurance fraud

- **Genomic Data Analysis**
  - Privacy breach for user and relatives
  - Protection required to avoid misdiagnosis

---

*Naperville Smart Meter Awareness v. City of Naperville, No. 16-3766 (7th Cir. 2018)*

*Hubaux et al., Genomic Data Privacy and Security: Where We Stand and Where We Are Heading* IEEE Security & Privacy 2017
Objectives of CRISP

We design and implement CRISP, a solution to compute securely on authenticated data at an acceptable cost in utility, without compromising privacy.

Privacy
Prevent the service provider from gaining more information than required.

Integrity
Prevent the user from cheating the service provider.

Utility
Ensure good quality of service for both user and service provider.

Deployability
Ensure smooth deployment with existing software and hardware infrastructure.
CRISP Overview

Utility
- Flexible Computations

Privacy
- Input Data Confidentiality

Security
- Data Authenticity

Fully Homomorphic Encryption (a)
- Enable polynomial computations on the encrypted data without decryption

Commitments and Blindings (c)
- Reveal only the result of the computation and prevent cheating

Zero-Knowledge Circuit Evaluation (b)
- Evaluate a tailored circuit checking the encryption and hash of the data

---

(a) Cheon et al., "Homomorphic Encryption for Arithmetic of Approximate Numbers", ASIACRYPT 2017
(b2) Chase et al., "Post-Quantum Zero-Knowledge and Signatures from Symmetric-Key Primitives", CCS 2017
(c) Baum et al., "More Efficient Commitments from Structured Lattice Assumptions", ePrint 2016
CRISP Model

Service Provider
User
Data Source

COLLECTION
TRANSFER
VERIFICATION
COMPUTATION
DATA RELEASE

SHA-256 based signature

F(\(\text{壑}\))=m

Honest
Malicious but Rational
Honest-but-curious

CRISP Model

Honest
Honest-but-curious
Malicious but Rational

SHA-256 based signature

F(\(\text{壑}\))=m

6
The user generates a proof guaranteeing
- Correct encryption
- Integrity of the data

We designed a custom circuit to check simultaneously
- The hash of the data
- The norm of the encryption noises
- The encryption of the data

CRISP relies on a Zero-Knowledge Circuit Evaluation and Lattice-Based Commitments

Eventually, the user sends the message $M=\{\text{ciphertexts, commitments, proof, hashes, signatures}\}$
Verification and Computation Phase

- **Data Source**
  - Honest-but-curious

- **User**
  - Malicious but Rational

- **Service Provider**
  - Honest-but-curious

**SHA-256 based signature**

**COLLECTION**

**TRANSFER**

**VERIFICATION**

**COMPUTATION**

**DATA RELEASE**

- Run the ZKCE verification protocol
- Compute homomorphically on the ciphertexts

\[ F(\text{file}) = m \]
The user and service provider engage in an interactive protocol to reveal only the result of the computation in a tamper-proof manner.

Two-round protocol the security of which relies on the security of the underlying commitment scheme.
Evaluation: Overhead

We evaluated the communication and computation overhead. Overall, we achieve:

- One-shot generation and offloading of the proof for multiple computations
- Packing of numerous data points into one ciphertext
- Support for several subsequent operations at no additional proof cost

<table>
<thead>
<tr>
<th>Use Case</th>
<th>$t_{\text{prove}}$ (s)</th>
<th>$t_{\text{verify}}$ (s)</th>
<th>Proof Size (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smart Metering</td>
<td>200 ± 10</td>
<td>82 ± 5</td>
<td>650</td>
</tr>
<tr>
<td>Disease Susceptibility based on Genomic Data</td>
<td>26 ± 4</td>
<td>13 ± 2</td>
<td>54</td>
</tr>
<tr>
<td>Location-Based Activity-Tracking</td>
<td>470 ± 40</td>
<td>210 ± 10</td>
<td>1,603</td>
</tr>
</tbody>
</table>

Acceptable overhead, considering:
- Possibility to offload the proof to a public billboard
- Several optimisations can reduce the overhead
Optimizations

Several optimizations are possible to reduce the proof size:

- **Zero-Knowledge Circuit Evaluator**

  **Preprocessing**
  Rely on an offline phase to reduce the number of required online interactions

- **Batching**
  Use the signature to its full extent to pack as many data points as possible (data source modification required)

- **Random Integrity Checks (RIC)**
  The verifier picks at random a subset of data points which authenticity is checked

---

(d) Katz et al. "Improved non-interactive zero knowledge with applications to post-quantum signatures." CCS 2019
Conclusion on CRISP

Reconciles **security**, **privacy**, and **utility** using zero-knowledge proofs and homomorphic encryption:

- Ensures **computations on authenticated data**
- Preserves privacy of the data
- Does not affect accuracy more than the FHE scheme
- **One-time communication** overhead for the prover

Future work

- Malicious Service Provider
- Further reduce the proof size relying on alternative zero-knowledge proofs or increasing the number of parties
Thank you

sylvain.chatel@epfl.ch

https://github.com/ldsec/CRISP